# UNIVERSITY OF ENGINEERING & MANAGEMENT,JAIPUR

## Lecture-wise Plan

**Subject Name: Mathematics-III** **Subject Code-M301**
**Year: 2nd Year** **Semester: Third**

| Module Number | Topics | Number of Lectures |
|---|---|---|
| 1 | **Fourier Series & Fourier Transform** | **8L** |
| | Introduction | 1 |
| | Fourier Series for functions of period 2π, Fourier Series for functions of period 2L | 3 |
| | Fourier Integral Theorem, Fourier Transform of a function, Fourier Sine and Cosine Integral Theorem. | 1 |
| | Properties of Fourier Transform, Fourier Transform of Derivatives. | 1 |
| | Convolution Theorem, Inverse of Fourier Transform. | 2 |
| 2 | **Introduction to Functions of a Complex Variable & Conformal Mapping, Complex Integration, Residue & Counter Integration** | **8L** |
| | Complex functions, Limit, Continuity and Differentiability, Analytic functions | 1 |
| | Cauchy-Riemann Equations, Harmonic function and Conjugate Harmonic function | 1 |
| | Construction of Analytic functions: Milne Thomson method. | 1 |
| | Simple curve, closed curve, smooth curve & contour, complex Integrals. | 1 |
| | Cauchy’s theorem, Cauchy-Goursat theorem, Cauchy’s integral formula, Cauchy’s integral formula | 2 |
| 3 | **Basic Probability Theory, Random Variable & Probability Distributions. Expectation** | **12L** |
| | Introduction | 1 |
| | Conditional probability, Independent events & Multiplication Rule. | 1 |
| | Baye’s theorem | 1 |
| | Random variable | 1 |
| | Probability density function & probability mass function. | 2 |
| | Expectation & Variance | 1 |
| | Binomial & Poisson distributions and related problems. | 2 |
| | Uniform, Exponential, Normal distributions and related problems. | 3 |
| 4 | **Partial Differential Equation (PDE) and Series solution of Ordinary Differential Equation (ODE)** | **7L** |
| | Origin of PDE, its order and degree, concept of solution in PDE. | 1 |
| | Different methods: Separation of variables, Laplace & Fourier transform methods. | 3 |
| | PDE I: One dimensional Wave equation. | 1 |
| | PDE II: One dimensional Heat equation | 1 |
| | PDE III: Two dimensional Laplace equation. | 1 |

**Assignment:**
**Module-1:**

1. Write the statement of Fourier integral Theorem.
2. If the Fourier series of function $f(x)$ is given by $a_0+\sum_{n=1}^{\infty} a_n \cos nx+\sum_{n=1}^{\infty} b_n \sin nx$, then $a_n$ is given by?
3. If the Fourier series of function $f(x)$ is given by $a_0+\sum_{n=1}^{\infty} a_n \cos nx+\sum_{n=1}^{\infty} b_n \sin nx$, then $b_n$ is given by?
4. If the Fourier series of function $f(x)$ is given by $a_0+\sum_{n=1}^{\infty} a_n \cos nx+\sum_{n=1}^{\infty} b_n \sin nx$, then $a_n$ is given by?
5. If the Fourier series of function $f(x)$ is given by $a_0+\sum_{n=1}^{\infty} a_n \cos nx+\sum_{n=1}^{\infty} b_n \sin nx$, then $b_n$ is given by?
6. If $F(p)$ is the Fourier transform of $f(x)$, then the Fourier transform of $f(ax)$ is given by?
7. If $F(p)$ is the Fourier transform of $f(x)$, then the Fourier transform of $f(x-a)$ is given by?
8. If $F(p)$ is the Fourier transform of $f(x)$, then the Fourier transform of $f(ax)$ is given by?
9. If $F(p)$ is the Fourier transform of $f(x)$, then the Fourier transform of $f(x-a)$ is given by?
10. Define periodic function
11. Define even function
12. Write the relation between two orthogonal functions.
13. IF convolution of two functions exists then the value of $F\left\{\frac{1}{\sqrt{2®}}\int_{-\infty}^{\infty} f(u)g(x-u)du; p\right\}=$
14. IF convolution of two functions exists then the value of $F\left\{\frac{1}{\sqrt{2®}}\int_{-\infty}^{\infty} f(u)g(x-u)du; p\right\}=$
15. IF convolution of two functions exists then the value of $F\left\{\frac{1}{\sqrt{2®}}\int_{-\infty}^{\infty} f(u)g(x-u)du; p\right\}=$
16. IF convolution of two functions exists then the value of $F\left\{\frac{1}{\sqrt{2®}}\int_{-\infty}^{\infty} f(u)g(x-u)du; p\right\}=$
17. Obtain the Fourier series for the function $f(x)=x^2, -® < x < ®$.
18. Obtain the fourier series for the function $f(x)=\frac{1}{4}\left(® - x^2\right), 0 < x < 2®$.
19. Obtain the fourier series for the function $f(x)=\sin ax, -® < x < ®$. $a$ being non-integer value.
20. Obtain the fourier series for the function $f(x)=x, -® < x < ®$.

**Module-2:**

21. Write Cauchy- Riemann equations for a function $f(z)=u(x,y)+iv(x,y)$.

22. Write necessary condition for a function $f(z) = u(x,y) + iv(x,y)$ to be analytic.
23. Write necessary and sufficient condition for a function $f(z) = u(x,y) + iv(x,y)$ to be analytic.
24. Write sufficient condition for a function $f(z) = u(x,y) + iv(x,y)$ to be analytic.
25. State Cauchy’s integral theorem.
26. Write Cauchy’s integral formula.
27. Write type of singularity of the function $\frac{\sin z}{z}$ at $z = 0$.
28. Write type of singularity of the function $\frac{z^3}{(z+1)^2(z-5)^4}$ at $z = 5$.
29. Write type of singularity of the function $\frac{1}{(z+1)^2(z-3)^2}$ at $z = -1$.
30. Write type of singularity of the function $\frac{z^2}{(z+1)(z-3)^2}$ at $z = 3$.
31. Examine that the function $f(x,y) = y^3 - 3x^2y$ is harmonic or not.
32. Examine that the function $f(x,y) = \frac{1}{2}\log(x^2+y^2)$ is harmonic or not.
33. Examine that the function $f(x,y) = \frac{x-y}{x^2+y^2}$ is harmonic or not.
34. Examine that the function $f(x,y) = 2x(1-y)$ is harmonic or not.
35. Evaluate $\int_0^{1+i} z^2 dz$, where $z$ is complex number.
36. Evaluate $\int_0^{1+2i} (1+z^2) dz$, where $z$ is complex number.
37. Evaluate $\int_0^{2+i} e^z dz$, where $z$ is complex number.
38. Evaluate $\int_0^{1+i} (z^2+3z+2) dz$, where $z$ is complex number.
39. Find the residue at the poles of $f(z) = \frac{\cot\text{®} z}{(z-a)^2}$.
40. Find the residue at the poles of $f(z) = \frac{z^2-2z}{(z+1)^2(z^2+4)}$.
41. Find the residue of $f(z) = \frac{z^3}{z^2-1}$ at $z = \infty$.
42. Find the residue of $f(z) = \frac{e^z}{z \sin mz}$ at $z = 0$.

**Module-3:**

1. If for two events $A$ and $B$ we have the following probabilities:

$P(A)=P(A|B)=\frac{1}{4};P(B|A)=\frac{1}{2}.$ Then check $A$ and $B$ are independent or not.

2. If $P(A\cap B)=\frac{1}{2},P(\overline{A}\cap\overline{B})=\frac{1}{2}$ and $2P(A)=P(B)=p,$ then find the value of $p.$

3. If for two events $A$ and $B$ we have the following probabilities:

   $P(A)=P(A|B)=\frac{1}{4};P(B|A)=\frac{1}{2}.$ Then find $P(\overline{A}|B)=.$

4. If $P(A\cap B)=\frac{1}{2},P(\overline{A}\cap\overline{B})=\frac{1}{3}$ and $P(A)=P(B)=p,$ then find the value of $p.$

5. If $A$ and $B$ are any two events and $P(A)=p_1;P(B)=p_2;P(A\cap B)=p_3.$ Then $P(\overline{A}\cup\overline{B})=$

6. If $A$ and $B$ are any two events and $P(A)=p_1;P(B)=p_2;P(A\cap B)=p_3.$ Then $P(\overline{A\cup B})=$

7. If $A$ and $B$ are any two events and $P(A)=p_1;P(B)=p_2;P(A\cap B)=p_3.$ Then $P(\overline{A}\cap\overline{B})=$

8. If $A$ and $B$ are any two events and $P(A)=p_1;P(B)=p_2;P(A\cap B)=p_3.$ Then $P(\overline{A}\cup B)=$

9. State Baye's theorem for mutually disjoint events.

10. If $f(x)=\begin{cases}ke^{-2x} & x\geq 0\\ 0 & otherwise\end{cases}$, then what will be the value of $k$ for which $f(x)$ be probability density function?

11. If $f(x)=\begin{cases}x & 0<x<1\\ 0 & otherwise\end{cases}$, then $f(x)$ is probability density function or not?

12. If $f(x)=\begin{cases}ke^{-x} & x\geq 0\\ 0 & otherwise\end{cases}$, then what will be the value of $k$ for which $f(x)$ be probability density function?

13. If $f(x)=\begin{cases}k\left(1-e^{-x}\right)^2 & x\geq 0\\ 0 & otherwise\end{cases}$, then what will be the value of $k$ for which $f(x)$ be probability density function?

14. Write the formula for mathematical expectation of a discrete random variable $X$ with probability mass function $f(x).$

15. Write the formula for mathematical expectation of a continuous random variable $X$ with probability density function $f(x).$

16. Write the formula for mathematical expectation of a discrete random variable $X$ with probability mass function $f(x).$

17. Write the formula for mathematical expectation of a continuous random variable $X$ with probability density function $f(x).$

18. A card is drawn from pack of 52 cards, find the probability of getting a king or a heart or a red card?
19. A card is drawn from a pack of 52 cards, if the value of faces cards 10, aces cards 1 and other according to denomination, find the expected value of the no. of point on the card.
20. A bag contains 10 red and 15 white balls. Two balls are drawn in succession. What is the probability that one of them is white and other red?
21. State Bayes' theorem.
22. A and B take turns in throwing two dice on the understanding that the first to throw 9 will be awarded a prize. If A has the first turn, show their respective chances of winning are in the ratio 9 : 8.
23. Three groups of children contain respectively 3 girls and 1boy; 2 girls and 2 boys; 1 girls and 3 boys, One child is selected at random from each group. Find the chance of selecting 1 girl and 2 boys.
24. A manufacturer supplies quarter horsepower motors in lots of 25. A buyer, before taking a lot, tests at random a sample of 5 motors and accepts the lot if they are all good; otherwise he rejects the lot. Find the probability that :(i) he will accept a lot containing 5defective motors ;(ii) he will reject a lot containing only one defective motors.
25. In an examination with multiple-choice questions, each question has four, out of which one is correct. A candidate ticked the answer either by his skill or by copying from his neighbours, The probability of guess is 1/3, copying is 1/6. The probability of correct answer by copying is 1/8. If a candidate answers a question correctly find the probability that he know the answer.
26. An urn contains 10 white and 3 black balls. Another urn contains 3 white and 5 black balls. Two balls are drawn at random from first urn and placed in the second urn and then one ball is taken at random from the latter. What is the probability that it is a white ball ?
27. Define the random variable, Explain the types of random variable with example.
28. A can hit a target 4 times in 7 shots, B 3 times in 5 shots and C three times in 5 shots. All of them fire one shot each simultaneously at the target. What is the probability that (i) 2 shots hit (ii) At least two shots hit ?
29. The probability that a student A solves a mathematics problem is 2/5 and the probability that a student B solves the problem is 2/3. What is the probability that (a) the problem is not solved (b) the problem is solved (c) both A and B solve the problem.
30. A company has four production section $S_1$, $S_2$, $S_3$& $S_4$ which contribute 30%, 20% 22% & 28% respectively produced 1%, 2%, 3% & 4% defective units, if a small unit is selected random & found to be defective, what is the probability that the unit selcected has came from (a) Section $S_1$ (b) Section $S_4$
31. From a city population, the probability of selecting a male or a smoker is 7/10, a male smoker is 2/5 and a male if a smoker is already selected is 2/3, find the probability of selecting (a) non-smoker (b) a male (c) a smoker if a male is first selected.
32. There are two bags A and B. A contains n white and 2 black balls & B contains 2 white and n black balls, one of the two bags is selected at random and two balls are drawn from it without replacement. If the both balls are drawn are white and the probability that the bag A was used to drawn the ball is 6/7. Find the value of n.

**Module-4:**

1. Bessel function of order $p=\pm\frac{1}{2}$, show that $J_{1/2}(x)=\sqrt{2/®x}\sin x$ and $J_{-1/2}(x)=\sqrt{2/®x}\cos x$.
2. Determine the order p of the following Bessel equation:

   a) $x^2y^{"}+xy'+(x^2-9)y=0$

b) $x^2 y^{"} + xy' + x^2 y = 0$

3. Solve the following heat flow problem:

$$\frac{\partial u}{\partial t} = 7\frac{\partial^2 u}{\partial x^2} \quad , \quad 0 < x < \text{®} \quad , \quad t > 0.$$

$$u(0,t) = u(\text{®}\ ,t) = 0\ , \quad t > 0\ ,$$

$$u(x,0) = 3\sin 2x - 6\sin 5x \quad , \quad 0 < x < \pi.$$

4. Prove that F satisfies the Laplace’s equation: $F = Cz^n$

$$\nabla^2 F = \frac{\partial^2 F}{\partial x^2} + \frac{\partial^2 F}{\partial y^2} = 0$$

# UNIVERSITY OF ENGINEERING & MANAGEMENT,JAIPUR

## Lecture-wise Plan

**Subject Name: Data Structure & Algorithm** **Subject Code-CS(ME)301**
**Year: 2$^{nd}$ Year** **Semester: Third**

| Module Number | Topics | Number of Lectures |
|---|---|---|
| 1 | **Introduction:** | **5L** |
| | 1. Why we need data structure? Concepts of data structures: a) Data and data structure b) Abstract Data Type and Data Type. Algorithms and programs, basic idea of pseudo-code.<br>. | 1 |
| | 2. Algorithm efficiency and analysis, time and space analysis of algorithms – order notations. | 4 |
| | **Linear data structure:** | |
| 2 | **Array:** | **2L** |
| | 1. Different representations – row major, column major. Sparse matrix - its application and usage. Array representation of polynomials. | 2 |
| 3 | **Linked List:** | **7L** |
| | 1. Singly linked list, circular linked list, doubly linked list, linked list representation of polynomial and applications. | 7 |
| 4 | **Stack and Queue:** | **6L** |
| | 1. Stack and its implementations (using array, using linked list), applications. | 2 |
| | 2. Queue, circular queue, dequeues. Implementation of queue- both linear and circular (using array, using linked list), applications. | 4 |
| 5 | **Recursion:** | **3L** |
| | 1. Principles of recursion – use of stack, differences between recursion and iteration, tail recursion. | 1 |
| | 2. Applications - The Tower of Hanoi, Eight Queens Puzzle. | 2 |
| | **Non Linear data structure:** | |
| 6 | **Trees:** | **8L** |
| | 1. Basic terminologies, forest, tree representation (using array, using linked list). Binary trees - binary tree traversal (pre-, in-, post- order), threaded binary tree (left, right, full) - non-recursive traversal algorithms using threaded binary tree, expression tree. | 4 |
| | 2. Binary search tree- operations (creation, insertion, deletion, searching). Height balanced binary tree – AVL tree (insertion, deletion with examples only). B- Trees – operations (insertion, deletion with examples only | 4 |
| | **Graphs:** | **5L** |
| | 1. Graph definitions and concepts (directed/undirected graph, weighted/un-weighted edges, sub-graph, degree, cut- | |

| | | |
|---|---|---|
| 7 | vertex/articulation point, pendant node, clique, complete graph, connected components – strongly connected component, weakly connected component, path, shortest path, isomorphism). Graph representations/storage implementations – adjacency matrix, adjacency list, adjacency multi-list. | 1 |
| | 2. Graph traversal and connectivity – Depth-first search (DFS), Breadth-first search (BFS) – concepts of edges used in DFS and BFS (tree-edge, back-edge, cross-edge, forward-edge), applications. | 2 |
| | **3.** Minimal spanning tree – Prim’s, Kruskal and Dijkstra algorithm (basic idea of greedy methods). | 2 |
| 8 | **Sorting, Searching and Hashing Technique:** | |
| | **Sorting Algorithms:** | 6L |
| | Bubble sort and its optimizations, insertion sort, shell sort, selection sort, merge sort, quick sort, heap sort (concept of max heap, application – priority queue), radix sort. | 6 |
| | **Searching:** | 2L |
| | Sequential search, binary search, interpolation search. | 2 |
| | **Hashing:** | 2L |
| | Hashing functions, collision resolution techniques. | 2 |
| **Total Number Of Hours = 46** | | |

Faculty In-Charge HOD, CSE Dept.

**Assignment:**

**Module-1(Introduction):**

**1.** DefineAbstractData Type, big oh, big omega, theta notationoftimecomplexity.

2. Findthetotalfrequency count of following code.

```
for send=1ton do
        forreceive=1tosenddo
                forack=2toreceivedo
                        message=send-(receive+ack)
                        ack=ack-1
                        send=send+1
                end
        end
end
```

**Module-2 (Linear data Structure):**

1. Write a function to insert a element after 4th position in an array.
2. Write a function to insert a element before 4th position in a single linked list
3. Write a function to insert a element after a particular data element 4 in a doubly linked list.
4. Write a function to concatenate two circular linked list.

5. Write a function to implement stack and queue using linked list.
6. Convert infix to prefix and postfix.
   A+B+C-D/E*R(S*T)/W+G
7. Define tail and tree recursion, explain them with example.

**Module-3(Non-linear data structure):**

1. Why AVL tree is required?
2. Construct the AVL tree.
   B,D,A,G,H,R,J,T,C,Y,X
3. Write a short note on B-Tree.
4. Write an algorithm of DFS and Dijkstra algorithm.

**Module-4(Sorting, Searching and Hashing):**

1. Explain quick and radix sort with example.
2. Why binary search is better than linear search.
3. Write down different techniques of collision resolution techniques.

# UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

## Lecture-wise Plan

Subject Name: **Basic Environmental Engineering** Subject Code: **CH-301**
Year: **2nd Year** Semester: **Third**

| Module Number | Topics | Number of Lectures |
|---|---|---|
| 1 | **Chapter 1: General** | **6L** |
| | 1. Basic ideas of environment, basic concepts, man, society & environment, their interrelationship. | 1L |
| | 2. Mathematics of population growth and associated problems, Importance of population study in environmental engineering,<br>definition of resource, types of resource, renewable, non-renewable, potentially renewable, effect of excessive use vis-à-vis | 2L |
| | 3. Materials balance: Steady state conservation system, steady state system with non conservative pollutants, step function. | 1L |
| | 4. Environmental degradation: Natural environmental Hazards like Flood, earthquake, Landslide-causes, effects and control/management; Anthropogenic degradation like Acid rain-cause, effects and control. Nature and scope of Environmental Science and Engineering. | 2L |
| | **Chapter 2: Ecology** | **6L** |
| | 1. Elements of ecology: System, open system, closed system, definition of ecology, species, population, community, definition of ecosystem-components types and function. | 1L |
| | 2. Structure and function of the following ecosystem: Forest ecosystem, Grassland ecosystem, Desert ecosystem, Aquatic ecosystems, Mangrove ecosystem (special reference to Sundar ban); Food chain [definition and one example of each food chain], Food web. | 2L |
| | 3. Biogeochemical Cycle- definition, significance, flow chart of different cycles with only elementary reaction [Oxygen, carbon, Nitrogen, Phosphate, Sulphur]. | 1L |
| | 4. Biodiversity- types, importance, Endemic species, Biodiversity Hot-spot, Threats to biodiversity, Conservation of biodiversity. | 2L |
| | **Chapter 3: Air pollution and control** | **7L** |
| | 1. Atmospheric Composition: Troposphere, Stratosphere, Mesosphere, Thermosphere, Tropopause and Mesopause | 1L |
| | 2. Energy balance: Conductive and Convective heat transfer, radiation heat transfer, simple global temperature model [Earth as a black body, earth as albedo], Problems. | 1L |
| | 3. Green house effects: Definition, impact of greenhouse gases on the global climate and consequently on sea water level, agriculture and marine food. Global warming and its consequence, Control of Global warming. Earth's heat budget. | 1L |
| | 4. Lapse rate: Ambient lapse rate Adiabatic lapse rate, atmospheric stability, temperature inversion (radiation inversion).<br>Atmospheric dispersion: Maximum mixing depth, ventilation coefficient, effective stack height, smokestack plumes and Gaussian plume model. | 1L |

| | | |
|---|---|---|
| | 5. Definition of pollutants and contaminants, Primary and secondary | 1L |
| | pollutants: emission standard, criteria pollutant.<br>Sources and effect of different air pollutants- Suspended particulate matter, oxides of carbon, oxides of nitrogen, oxides of sulphur, particulate, PAN. | |
| | 6. Smog, Photochemical smog and London smog.<br>Depletion Ozone layer: CFC, destruction of ozone layer by CFC, impact of other green house gases, effect of ozone modification. | 1L |
| 2 | 7. Standards and control measures: Industrial, commercial and residential air quality standard, control measure (ESP. Cyclone separator, bag house, catalytic converter, scrubber (ventury), Statement with brief reference). | 1L |
| | **Chapter 4: Water Pollution and Control** | **8L** |
| | 1. Hydrosphere, Hydrological cycle and Natural water. | 1L |
| | 2. Pollutants of water, their origin and effects: Oxygen demanding wastes, pathogens, nutrients, Salts, thermal application, heavy metals, pesticides, volatile organic compounds. | 2L |
| | 3. River/Lake/ground water pollution: River: DO, 5 day BOD test, Seeded BOD test, BOD reaction rate constants, Effect of oxygen demanding wastes on river[deoxygenation, reaeration], COD, Oil, Greases, pH. | 1L |
| | 4. Lake: Eutrophication [Definition, source and effect].<br>Ground water: Aquifers, hydraulic gradient, ground water flow (Definition only) | 1L |
| | 5. Standard and control: Waste water standard [BOD, COD, Oil, Grease], Water Treatment system [coagulation and flocculation, sedimentation and filtration, disinfection, hardness and alkalinity, softening]<br>Waste water treatment system, primary and secondary treatments [Trickling filters, rotating biological contractor, Activated sludge, sludge treatment, oxidation ponds] tertiary treatment definition. | 2L |
| | 6. Water pollution due to the toxic elements and their biochemical effects: Lead, Mercury, Cadmium, and Arsenic | 1L |
| 3 | **Chapter 5: Land Pollution** | **3L** |
| | 1. Lithosphere; Internal structure of earth, rock and soil | 1L |
| | 2. Solid Waste: Municipal, industrial, commercial, agricultural, domestic, pathological and hazardous solid wastes; Recovery and disposal method- Open dumping, Land filling, incineration, composting, recycling.<br>Solid waste management and control (hazardous and biomedical waste). | 2L |
| | **Chapter 5: Noise Pollution** | **2L** |
| | 1. Definition of noise, effect of noise pollution, noise classification [Transport noise, occupational noise, neighbourhood noise] | 1L |
| | 2. Definition of noise frequency, noise pressure, noise intensity, noise threshold limit value, equivalent noise level, *L*10 (18 hr Index) , *n Ld* , Noise pollution control. | 1L |
| | **Chapter 6: Environmental Management** | **2L** |
| | 1. Environmental impact assessment, Environmental Audit, Environmental laws and protection act of India, Different international environmental treaty/ agreement/ protocol. | 2L |
| **Total Number Of Hours = 34L** | | |

# UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

## Lecture-wise Plan

Subject Name: **Basic Environmental Engineering** Subject Code: **CH-301**
Year: **2$^{nd}$ Year** Semester: **Third**

Faculty In-Charge HOD, ME Dept.

**Assignment:**

**Module-1.**
**1.** Write short notes for the following:
(a) Flood (b) Landslides (b) Earthquake (c) Acid Rain

**2.** Suppose an anemometer at a height of 40 m above ground measure wind velocity =5.5 m/s. Estimate the wind speed at an elevation of 500 m in rough terrain if atmosphere is unstable (i.e., k = 0.2).

**Module-2.**
1. A BOD test is run using 50 ml of wastewater mixed with 100 ml of pure water. The initial DO of the mixture is 6 mg/l and after 5 days it becomes 2 mg/l. After a long time, the DO remains fixed at 1 mg/l.
(i)What is the 5 days BOD ($BOD_5$)?
(ii)What is the ultimate BOD ($BOD_u$)?
(iii)What is the remaining BOD after 5 days?
(iv)What is the reaction rate constant measured at 20°C?
(v)What would be the reaction rate if measured at 35°C?

2. Draw the flow diagram for the following (a) Surface water treatment (b) Waste water Treatment.

3. Draw the Oxygen sag curve.

**Module-3.**
**1. a)** If two machines produces sounds of 80 dB and 120 dB simultaneously, what will be the total sound level.
b) Calculate the intensity of 100 dB sounds.

2. Write a report on the environmental problems related to an abandoned airport. Mention various measures by which it can be used again for other purposes.

# UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

## Lecture-wise Plan

Subject Name: Applied Thermodynamics Subject Code-ME301
Year: 2nd Year Semester: Third

| Module Number | Topics | Number of Lectures |
|---|---|---|
| 1 | **Fundamentals of Thermodynamics** | **6L** |
| | Review of fundamentals of thermodynamics, its definition and application in engineering. Heat and work, First law for unsteady flow system | 1L |
| | Recapitulations of laws of thermodynamics, steady flow and non steady flow and its applications. | 2L |
| | Pure Substance and its example, Properties of pure substance; Phases of pure substances- Phase rule; Phase Change Processes of Pure Substances – triple point., critical point. Saturation temperature and pressure. | 1L |
| | Property diagrams of Phase change Processes of water on T-V, P-V, T-S and h-s plots ; P-V-T surface for phase change. Concept of dryness fraction or quality for liquid vapor mixture. | 2L |
| 2 | **2nd Law of Thermodynamics** | **6L** |
| | The 2nd Law of Thermodynamics; the corollaries & their proofs; the property of entropy; entropy change of a pure substance. | 2L |
| | Energy analysis | 1 L |
| | Tds equations and calculation of entropy change; concept and uses of entropy; The second law of thermodynamics for an open system and its numerical. | 2 L |
| | Concept of entropy generation, Reversible work and irreversibility and its numerical. | 1 L |
| 3 | **Joule Thompson Effect** | **4L** |
| | Maxwell relations, T-ds equations and its derivations. | 2L |
| | Derivations of Joule Thompson co-efficient & Clapeyron Equation. | 2L |
| 4 | **Study of Thermodynamic Cycles** | **9L** |
| | Introduction to I.C.Engine, Air Standard cycles. | 2L |
| | Otto cycle p-v and t-s diagram, efficiency and its numerical | 2L |
| | Diesel cycle p-v and t-s diagram , efficiency and its numerical. | 1L |
| | Dual Combustion cycle p-v and t-s diagram and its numerical. Different cycle performance comparison. | 2L |
| | Reciprocating air compressors; the compressor cycle with and without clearance.. | 1L |
| | Efficiencies; volumetric efficiency & its effect on performance. | 1L |
| 5 | **Vapor power cycles** | **6L** |
| | Vapor power cycles, Rankine cycle & its modifications its p-v,t-s, h-s diagram and efficiency calculations. | 2L |
| | Numerical on rankine cycle. | 1L |
| | Reheat & Regenerative cycle for steam. | 2L |

| | | |
|---|---|---|
| | Binary cycle and cogeneration. | 1L |
| 6 | **Refrigeration system and carnot cycle** | **10L** |
| | Introduction to Refrigeration systems, Refrigeration systems cycles. | 1L |
| | Belcolleman cycle, and its numerical. | 2L |
| | Reversed carnot cycle; components and analysis of simple vapour compression Refrigeration cycle. | 2L |
| | Numerical on simple vapour compression Refrigeration cycle. | 1L |
| | Actual Refrigeration cycles. Use of psychometric charts & processes for air conditioning. | 2L |
| | Numerical on Actual Refrigeration cycles using psychometric chart. | 1L |
| | Vapour Absorption Refrigeration cycle. | 1L |
| **Total Number Of Hours = 41** | | |

## Assignment No. 1

Q1. What do you understand by thermodynamic system? What is difference between a closed system & open system?

Q2. Distinguish between the terms ‘change of state’, ‘path’ and ‘process’

Q3. What is thermodynamic cycle?

Q4. What are thermodynamic properties? Distinguish between intensive and extensive property.

Q5. Explain Zeroth Law of thermodynamics.

Q6. Explain Quasistatic process. What is its characteristics feature?

Q7. Define Work and Heat transfer.

Q8. State 1st law of thermodynamics.

Q9. Show that energy is a property of a system.

Q10. What is the differences between heat and internal energy?

Q11. Define Enthalpy. What does Enthalpy of an ideal gas depends only on temperature?

Q12. Define $C_P$ and $C_V$ ?

Q13. What is PMM1? Why is it impossible?

Q14. Derive an expression for Steady flow energy equation for a single stream of fluid entering and leaving the control volume.

Q15. Define

1. Steady flow process.

2. Throttling process.
3. Free expansion process.

**Assignment No. 2**

Q1. What is pure a substance.

Q2. What do you understand by triple point?

Q3. What do you understand by degree of superheating and degree of super cooling?

Q4. Derive the P.V diagram for pure substance.

Q5. Find the enthalpy and entropy and volume of steam at 1.4 MPa, 380$^0$C.

Q6. What is quality of steam? Explain dryness fraction.

Q7. Find the enthalpy & entropy of steam when the pressure is 2MPa and specify volume is 0.09 m$^3$/kg.

Q8. Prove that for a mixture of liquid and vapour the specific volume is given as

$v = v_f + x( v_g - v_f)$.

*Where $x$ =quality of mixture, $v_f$ =specific volume liquid, $v_g$= specific volume vapour*

Q9. Steam initially at 0.3 MPa and 250˚C is cooled at constant volume, Determine the following by parameter at different conditions of steam with drawing a neat p-v diagram.

(a) At what temperature will the steam become saturated vapour ?

(b) What is the quality at 80˚C?

(c)What is the heat transferred per kg of steam in cooling from 250˚C to 80˚C?

**Assignment No. 3**

Q1. What is 2$^{nd}$ law of thermodynamic? Give the Kelvin Planck statement of 2$^{nd}$ law.

Q2. What is thermal energy reservoir? Explain the terms ‘source’ and ‘sink’.

Q3. What is PPM2? Why is it impossible?

Q4. What is Refrigerator? Define its COP.

Q5. What is heat pump?

Q6. Define Entropy. Explain the principal of Entropy increase.

Q7. A carnot engine absorbs 200J of heat from a reservoir at the temp. of the normal boiling point of

Water and rejects heat to a reservoir at the temp. of triple point of water. Find the heat rejected, the work done by the engine and the thermal efficiency.

Q8. A cyclic heat engine works between a source temperature of $800^o$ C and sink temperature of $30^o$ C. What will be the least rate of heat rejection per KW net output of the engine?

Q9.Write down the *Inequality of Clausius* and give the criteria of reversibility, irreversibility and impossibility of a thermodynamic cycle.

## Assignment No. 4

Q1. Drive Maxwell's equation.
Q2. Write down the 1st & 2nd Tds equations.
Q3. Explain Joule Kelvin effect.
Q4. What is inversion temperature?
Q5. Derive the expression for the difference in heat capacities $C_P$ & $C_V$.
Q6. What is Joule Thomson Coefficient?

## Assignment No. 5

Q1. Derive an equation of air standard efficiency of following ideal gas power

cycles with P-V, T-S and H-S diagram: a. Carnot Cycle

b. Otto Cycle c. Diesel Cycle

d. Joule (Brayton) Cycle

Q2. Compare Otto, Diesel and Dual cycle for different aspects.
Q3. In an air standard diesel cycle, the compression ratio is 16, and at the beginning of isentropic compression, the temperature is 15˚C and the pressure is 0.1 MPa. Heat is added until the temperature at the end of the constant pressure process is 1480˚C. Calculate the following
(a) Cut off ratio (b) The heat supplied per kg of air (c) The cycle efficiency (d) The m.e.p.

Q4. An engine working on the Otto cycle is supplied with air at 0.1 Mpa, 35˚C . The compression ratio is 8. Heat supplied is 2100 kJ/kg. Calculate the maximum pressure and temperature of the cycle , the cycle efficiency , and the mean effective pressure . (For air, $C_P$= 1.005, $C_v$= 0.718 , R= 0.287 kJ/kg K.)

## Assignment No. 6

Q1. What do mean by refrigeration? and discuss the term tonne of refrigeration(TR).
Q2. Explain various types of refrigeration cycles.

# UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

## Lecture-wise Plan

Subject Name: Strength of Materials Subject Code-ME302
Year: 2nd Year Semester: Third

| Module Number | Topics | Number of Lectures |
|---|---|---|
| 1 | **Stress-strain** | **7L** |
| | Concept of stress, strain, types of stresses, elasticity, Hooke's law | 1L |
| | Stress-strain relationship, factor of safety, analysis of bars with different cross section | 1L |
| | Thermal stresses, analysis of bar of uniform strength | 1L |
| | Problem practice | 1L |
| | Strain, poisson's ratio, volumetric strain, bulk modulus | 1L |
| | Complementary shear stress, stresses on inclined section, diagonal stresses, modulus of rigidity | 1L |
| | Problem practice | 1L |
| 2 | **Principal stress-strain** | **6L** |
| | Principal planes and principal stresses, method of determining stresses on oblique section | 1L |
| | Mohr,s circle | 1L |
| | Strain on an oblique plain, Mohr's strain circle | 1L |
| | Problem practice | 1L |
| | Strain energy and impact loading | 1L |
| | Problem practice | 1L |
| 3 | **Shear force and, bending moment & bending stress** | **13L** |
| | Shear force & bending moment, types of beam and load, sign conventions | 1L |
| | Shear force & bending moment for cantilever beam | 1L |
| | Problem practice | 1L |
| | Shear force & bending moment for simply supported beam | 1L |
| | Problem practice | 1L |
| | Problem practice | 1L |
| | Shear force & bending moment for over-hanging beams | 1L |
| | Problem practice | 1L |
| | S.F & B.M for inclined load, beams subjected to couple | 1L |
| | Relationship of load, S.F, B.M and problem practice | 1L |
| | Bending stresses in beams | 1L |
| | Problem practice | 1L |
| | Combined direct and bending stresses, problem practice | 1L |

# UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

## Lecture-wise Plan

Subject Name: Strength of Materials Subject Code-ME302
Year: 2nd Year Semester: Third

| | | |
|---|---|---|
| 4 | **Torsion** | **5L** |
| | Shear stress in a circular shaft subjected to torsion, maximum torque, problem practice | 1L |
| | Torque for hollow circular shaft, power transmission | 1L |
| | Problem practice | 1L |
| | springs | 1L |
| | Problem practice | 1L |
| 5 | **Deflection of beam** | **6L** |
| | Deflection and slope and their relation with problems | 1L |
| | Deflection of S.S.B. for different conditions with problems | 1L |
| | Macaulay’s method, moment area method | 1L |
| | Problem practice | 1L |
| | Method of superposition, theorem of Castigliano | 1L |
| | Problem practice | 1L |
| 6 | **Column** | **5L** |
| | Failure of a column, Euler’s column theory, end conditions, crippling load expression, effective length | 1L |
| | Problem practice | 1L |
| | Rankine’s formula, straight line formula | 1L |
| | Problem practice | 1L |
| | Problem practice | 1L |
| **Total number of lectures = 42** | | |

**ASSIGNMENT 01**

1. Define longitudinal strain and lateral strain.
2. State Hooke’s law.
3. Define modular ratio, Poisson’s ratio
4. What is modulus of elasticity?
5. What is the use of Mohr’s circle?
6. What do you meant by stiffness?
7. Explain lateral strain with a neat sketch
8. What are principal planes?
9. Give the expression for major principal stress in a two dimensional system
10. What are the types of stresses developed in thin cylinders subjected to internal pressure?
11. Write the relationship between bulk modulus, rigidity modulus and Poisson’s ratio.

12. Draw stress – strain diagram for mild steel, brittle material and a ductile material and indicate salient points.
13. What is principle of super-position?
14. Draw the Mohr’s circle for a state of pure shear and indicate the principal stresses.
15. Differentiate thin cylinder & thick cylinder
16. What is the procedure for finding the thermal stresses in a composite bar?
17. Define the term ‘obliquity’ and how it is determined.
18. Define Factor of safety.
19. What do you meant by thermal stresses?
20. Define working stress & allowable stress

## ASSIGNMENT 02

1. What is the maximum bending moment for a simply supported beam subjected to uniformly distributed load and where it occurs?
2. Define shear stress.
3. What is shear force in a beam?
4. What is bending moment in a beam?
5. List the types of supports
6. Derive the relation between bending moment and shear force.
7. What is meant by section modulus?

8. (i) Derive an expression for bending moment equation

(ii) A rectangular beam 300 mm deep is simply supported over the span of 4 m. Determine the uniformly distributed load per metre which the beam may carry, if the bending stress should not exceed 120N/mm2. Take I=8×106 mm4. 9. A cantilever beam of 2 m long carries a uniformly distributed load of 1.5 kN/m over a length of 1.6 m from the free end. Draws shear force and bending moment diagrams for the beam.

9.. A simply supported beam 6 m long is carrying a uniformly distributed load of 5 kN/m over a length of 3 m from the right end. Draw shear force and bending moment diagrams for the beam and also calculate the maximum bending moment on the beam.

## ASSIGNMENT 03

1. A beam of length 6 m is simply supported at its ends and carries two point loads of 48 kN and 40 kN at a distance of 1 m and 3 m respectively from the left support. Find
(i) Deflection under each load

(ii) Maximum deflection
(iii) The point at which the maximum deflection occurs.
Take I=85×106 mm4 E = 2×105 N/mm2
2. A steel joist, simply supported over a span of 6 m carries a point load of 50 kN at 1.2 m from the left hand support. Find the position and magnitude of the maximum deflection. Take EI = 14 X 10 12 N/mm2

3. A cantilever of length 2.5m is loaded with an udl of 10 kN/m over a length 1.5m from the fixed end. Determine the slope and deflection at the free end. Determine the slope and deflection at the free end of the cantilever L = 9500cm4, E = 210 GN / m2 Using Moment area method.

## ASSIGNMENT 04

1. i) Derive the torsion equation for a circular shaft of diameter'd' subjected to torque 'T'.
ii) Find the torque that can be transmitted by a thin tube 6 cm mean diameter and wall thickness 1 mm. the permissible shear stress is 6000 N/cm2.
2. A close coiled helical spring is made of a round wire having 'n' turns and the mean coil radius R is 5 times the wire diameter. Show that the stiffness of the spring = 2.05 R/n. If the above spring is to support a load of 1.2kN with 120mm compression. Calculate mean radius of the coil and number of turns assuming G = 8200 N/mm2 and permissible shear stress, λ allowable = 250 N/mm2.
3. A steel shaft ABCD having a total length of 2400mm is contributed by three different sections as follows. The portion AB is hollow having outside and inside diameters 80mm and 50mm respectively, BC is solid and 80mm diameter. CD is also solid and 70mm in diameter. If the angle of twist is same for each section, determine the length of each portion and the total angle of twist. Maximum permissible shear stress is 50 MPa and shear modulus 0.82x105 MPa.
4. It is required to design a close coiled helical spring which shall deflect 1mm under and axial load of 100N at a shear stress of 90 MPa. The spring is to be made of round wire having shear modulus of 0.8x105 MPa. The mean diameter of the coil is to times that at the coil wire. Find the diameter and length of the wire.
5. A solid circular shaft transmits 75kW power at 200rpm. Calculate the shaft diameter, if the twist in the shaft is not to exceed one degree in 2m length of shaft and shear stress is not exceed 50 N/mm2. Assume the modulus of rigidity of the material of the shaft as 100 kN/mm2.

# UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

## Lecture-wise Plan

Subject Name: Strength of Materials Subject Code-ME302
Year: 2nd Year Semester: Third

### ASSIGNMENT 05

1.A rectangular block of material is subjected to a tensile stress of 110 N/mm2 on one plane and a tensile stress of 47 N/mm2 on the plane at right angle to the former. Each of the above stress is accompanied by a shear stress of 63 N/mm2 Find (i) The direction and magnitude of each of the principal stress (ii) Magnitude of greatest shear stress.

2.At a point in a strained material, the principal stresses are100 N/mm2 (T) and 40 N/mm2 (C). Determine the resultant stress in magnitude and direction in a plane inclined at 600 to the axis of major principal stress. What is the maximum intensity of shear stress in the material at the point?

# UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

## Lecture-wise Plan

Subject Name: Engineering Materials Subject Code-ME303
Year: 2nd Year Semester: Third

| Module Number | Topics | Number of Lectures |
|---|---|---|
| 1 | **Introduction to Engineering Materials** | **6L** |
| | Introduction to Material Science and Discussion about Course | 1L |
| | Crystal Structures and Lattices. | 1L |
| | Crystal Imperfections, Slip and Dislocations | 1L |
| | Elasticity and Plasticity, Plastic Deformations | 1L |
| | Hot and Cold Working of Metals, Recovery, Recrystallization and Grain Growth | 2L |
| 2 | **Phase Diagram** | **6L** |
| | What is a Phase Diagram and its importance | 1L |
| | Solidification and structures of metal and alloys | 1 L |
| | Iron Carbon Phase Diagram | 1 L |
| | Properties of metals – Mechanical, Physical and Electrical | 1 L |
| | Binary and Ternary phase Diagrams | 2 L |
| 3 | **Study of Microstructures** | **4L** |
| | Powder Metallurgy and its applications | 2L |
| | Fatigue, Creep, Slip and Dislocations | 2L |
| 4 | **Heat Treatment Procedures** | **5L** |
| | Annealing and types of annealing | 2L |
| | Tempering and Normalising | 1L |
| | Hardening and types of hardening process | 1L |
| | Various types of furnaces | 1 L |
| 5 | **Special types of steel** | **5L** |
| | Tool Steel and their classification | 2L |
| | Cast Iron and its classification | 2L |
| | Stress strain relationship and it properties | 1L |
| 6 | **Non Ferrous Metals and its types** | **6L** |
| | Copper and its types, Properties of copper | 1L |
| | Aluminium and its uses, types and its properties | 1L |
| | Alloys and effect of alloying in steel | 2L |
| | Semiconductors and Composites | 2L |
| **Total Number Of Hours = 32** | | |

## Assignment No. 1

1. What is LASER? Explain the principle of operation of He-Ne laser.
2. What are optical fibers? Discuss its principle of operation.
3. Discuss about the performance of metals & ceramics at high temp.
4. Differentiate between addition & condensation polymerization.
5. What are ferroelectric materials? Discuss about different types of ferro-electric materials.
6. What are the functions of matrix phase in a fiber- reinforced composite?
7. Derive an expression for electric polarization of di-electric in an electric field .
8. Explain the Ferro-electric phenomena w.r.t BaTiO3 .
9. Explain the methods of preventions of following types of corrosion:
a. dezincification
b. season cracking
c. caustic embrittlement of boilers
d. intergranular corrosion of steel.
10. Discuss about silicate structures.

## Assignment No. 2

11. Differentiate between type-1 & type-2 superconductors with examples.
12. Explain the principle of operation of Ruby laser.
13. Discuss the various applications of optical fiber.
14. Assume that electron is a small sphere of radius R, its charge & mass being distributed uniformly through out its volume. Derive an expression for its spin magnetic moment. [This model of the electron is too mechanistic to be in the spirit of quantum physics view of this particle].
15. Discuss the advantages & dis-advantages of classical free electron theory.
16. Explain how a four level laser system works?
17. Derive an expression for the electronic polarizability in terms of atomic radius.
18. Draw the polymer structure for polymethyl methacrylate & polyhexamethylene adipamide. Indicate whether they are thermoplastic or thermosetting polymers. Mention one application of each polymer.
19. Compare & describe the properties of GFRP & CFRP composites.
20. What is post tensioning in re-inforced concrete? How is it carried out? State at least two applications of this material.

## Assignment No. 3

21. State the mechanical properties of ceramics.
22. What is corrosion? How is it prevented for metallic materials?
23. How optical fibers are in use to improve the living conditions in the world?
24. Explain how failure models help the manufacturer to manufacture a product?
25. Derive an expression for the concentration of holes in the valence band of an intrinsic semiconductor.
26. Explain the mechanism of differential aeration of metals.Give two examples where differential aeration effects are seen.
27. What are the functions of the dispersed phase & matrix phase in composites?
28. What are the postulates of Drude –Lorentz theory of metals?
29. Give the mechanism of addition and condensation polymerization. What is the minimum functionality required for a monomer to form a cross- linked polymer?
30. Describe the steps to be adopted in the production of bricks for
(a) Civil construction work

(b)Refractory linings.

**Assignment No. 4**

31. Give the comparison between ferromagnetic, ferromagnetic and ant ferromagnetic substance.
32. Enumerate the potential use of composite materials.
33. Menntion few examples to highlight the applications of Ceramics in medical science, interior decoration and in electronic gadgets.
34. Derive an expression for the numerical aperture of a step index optical fibre in terms of refractive indices of core and cladding.
35. Enumerate the different methods to control or to prevent the corrosion of metals.
36. Discuss various characteristics of ceramics of ceramic white ware products and their industrial applications.
37.Derive an expression for the thermal conductivity of a material in terms of mean free path average speed of electrons.
38. What are the advantages and dis-advantages of plastics?
39. What is R.C.C? Give its composition and strength. How can you make it water proof?
40. Compare and describe the properties of GFRP and CFRP composites.

**Assignment No. 5**

41. Give a comparison between hard and soft ferromagnetic substance.
42. Differentiate between addition and condensation polymerization. Why does natural rubber need vulcanization? What is injection moulding of plastics?
43. Calculate the maximum percentage of sulpher that can be present vulcanized rubber.
44. Explain pitting corrosion. Discuss important methods for corrosion control.
45. Mention mechanical properties of ceramics. What are its important applications?
46. Find the number-average molecular weight of a sample of PTEE for which the number average degree of polymerization is 296.
47. What is Hall Effect? How you will determine the mobility of electrons in germanium knowing only the resistivity & Hall-coefficient of it.
48. The fermi energy in copper at 0k on the assumption that each copper atom contributes one electron to the electron gas is 7.04 ev .Calculate the Fermi energy & average energy of an electron in the metal at 300k.
49. Explain how failure analysis of material selection helps the manufacturer to manufacture a better product.
50. What is acceptance angle in optical fiber? Derive an expression for the numerical aperture of a step-index optical fiber in terms of its refractive indices of core & cladding.

# UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

## Lab Manual

**Title of Course: Technical Report Writing & Language Lab**
**Course Code: HU481**
**L-T-P scheme: 0-0-2** **Course Credit: 2**

**Objectives:**
1. To inculcate a sense of confidence in the students.
2. To help them become good communicators both socially and professionally.
3. To assist them to enhance their power of Technical Communication.

**Learning Outcomes:**

**Course Contents:**
**Exercises that must be done in this course are listed below:**
Exercise No.1: Report Types (Organizational/Commercial/Business/Project)

Exercise No. 2: Report Format &Organization of Writing Materials

Exercise No. 3: Report Writing (Practice Sessions &Workshops)

Exercise No. 4: Introductory Lecture to help the students get a clearidea of Technical Communication& the need of Language Laboratory Practice Sessions

Exercise No. 5: Conversation Practice Sessions: (To be done as real life interactions)
a) Training the students by using Language Lab Device/Recommended Texts/cassettes/cd to get their Listening Skill & Speaking killhoned
b) Introducing Role Play & honing overall Communicative Competence

Exercise No. 6: Group Discussion Sessions:
a) Teaching Strategies of Group Discussion
b) Introducing Different Models & Topics of Group Discussion
c) Exploring Live/Recorded GD Sessions for mending students' attitude/approach & for taking remedial measure Interview Sessions;
d) Training students to face Job Interviews confidently and successfully
e) Arranging Mock Interviews and Practice Sessions for integrating Listening Skill with Speaking skill in a formal situation for effective communication

Exercise No. 7: Presentation:
a) Teaching Presentation as a skill
b) Strategies and Standard Practices of Individual/Group Presentation
c) Media & Means of Presentation: OHP/POWERPOINT/Other Audio-Visual Aids

Exercise No. 8: Competitive Examination:
a) Making the studentsawareofProvincial/National/InternationalCompetitiveExaminations
b) Strategies/Tactics for success in Competitive Examinations
c) SWOT Analyse is and its Application in fixing Target

**Text Book:**
1. Nira Konar: English Language Laboratory: A Comprehensive Manual

D. Sudharani: Advanced Manual for Communication Laboratories& Technical Report Writing
Pearson Education (W.B.edition), 2011PHI Learning,2011

# UNIVERSITY OF ENGINEERING AND MANAGEMENT, JAIPUR

# Lab Manual

**Title of Course: Data Structure & Algorithm Lab**
**Course Code: CS(ME)391**
**L-T-P scheme: 0-0-3** **Course Credit: 2**

**Objectives:**

1. Develop problem solving ability using Programming.
2. Develop ability to design and analyze algorithms.
3. Introduce students to data abstraction and fundamental data structures.
4. Develop ability to design and evaluate Abstract Data Types and data structures.
5. Apply data structure concepts to various examples and real life applications

**Learning Outcomes:**

The course will use hands on practice and applying the knowledge gained in theory course to different day to day real world applications..Upon the completion of data structure and algorithm practical course, the student will be able to:

- **Understand** and implement different type of data structure techniques
- **Analyze** the hashing method.
- **Implement** different type of sorting searching techniques.

**Course Contents:**

**Exercises that must be done in this course are listed below:**

Exercise No.1: Implementation of array operations
Exercise No. 2: Implementation of linked lists: inserting, deleting a linked list.
Exercise No. 3: Stacks and Queues: adding, deleting elements
Exercise No. 4: Evaluation Problem: Evaluation of infix to postfix expressions on stack.
Exercise No. 5: Circular Queue: Adding & deleting elements
Exercise No. 6: Implementation of stacks using linked lists, Polynomial addition, Polynomial multiplication
Exercise No. 7: Sparse Matrices: Multiplication, addition.
Exercise No. 8: Recursive and Non-recursive traversal of Trees
Exercise No. 9: Threaded binary tree traversal. AVL tree implementation
Exercise No. 10: Application of sorting and searching algorithms

**Text Book:**

1. Yashavant Kanetkar, Abduln A.P.J. Kalam,” Data Structure Through C”,2nd edition, BPB Publications
2. Seymour Lipschutz,“Data Structures**”,**Revised First edition,McGraw Hill Education.

**Recommended Systems/Software Requirements:**

**1.** Intel based desktop PC with minimum of 166 MHZ or faster processor with at least 64 MB RAM and 100 MB free disk space.
**2.** Turbo C or TC3 complier in Windows XP or Linux Operating System.

## Exercise No.1: Implementation of array operations

**Description:**

An array is a collection of similar data elements. These data elements have the same data type.The elements of the array are stored in consecutive memory locations and are referenced by an `index`(also known as the subscript). The subscript is an ordinal number which is used to identify an element of the array.There are a number of operations that can be performed on arrays. These operations include:

KATraversing an array
2) Inserting an element in an array
MASearching an element in an array
NADeleting an element from an array

QASorting an array in ascending or descending order

**Aim: Write a program to insert a number at a given location in an array.**

**Algorithm:**

The algorithm `INSERT` will be declared as `INSERT(A,N,POS,VAL)`. The arguments are
 Step1: `A`, the array in which the element has to be inserted
 Step2: `N`, the number of elements in the array
 Step3: `POS`, the position at which the element has to be inserted
 Step4: `VAL`, the value that has to be inserted

**Program:**

```
#include <stdio.h>
#include <conio.h>
int main()
{
 int i, n, num, pos, arr[10];
 clrscr();
 printf("\n Enter the number of elements in the array : ");
 scanf("%d", &n);
 for(i=0;i<n;i++)
 {
  printf("\n arr[%d] = ", i);
  scanf("%d", &arr[i]);
 }
 printf("\n Enter the number to be inserted : ");
 scanf("%d", &num);
 printf("\n Enter the position at which the number has to be added : scanf("%d", &pos);
 for(i=n–1;i>=pos;i—)
  arr[i+1] = arr[i];
 arr[pos] = num;
 n = n+1;
 printf("\n The array after insertion of %d is : ", num);
 for(i=0;i<n;i++)
 printf("\n arr[%d] = %d", i, arr[i]);
 getch();
 return 0;
}
```

**Input:**

```
 Enter the number of elements in the array : 5
 arr[0] = 1
 arr[1] = 2
 arr[2] = 3
 arr[3] = 4
 arr[4] = 5
 Enter the number to be inserted : 0
 Enter the position at which the number has to be added : 3
```

**Output:**

```
 The array after insertion of 0 is :
 arr[0] = 1
 arr[1] = 2
 arr[2] = 3
 arr[3] = 0
 arr[4] = 4
 arr[5] = 5
```

**Aim:Write a program to delete a number from a given location in an array.**

**Algorithm:**

The algorithm `DELETE` will be declared as `DELETE(A, N,POS)`. The arguments are:

Step2: N, the number of elements in the array
Step3: POS, the position from which the element has to be deleted

**Program**

```
#include <stdio.h>
#include <conio.h>
int main()
{
 int i, n, pos, arr[10];
 clrscr();
 printf("\n Enter the number of elements in the array : ");
 scanf("%d", &n);
 for(i=0;i<n;i++)
 {
  printf("\n arr[%d] = ", i);
  scanf("%d", &arr[i]);
 }
 printf("\nEnter the position from which the number has to be deleted : ");
 scanf("%d", &pos);
 for(i=pos; i<n–1;i++)
  arr[i] = arr[i+1];
 n—;
 printf("\n The array after deletion is : ");
 for(i=0;i<n;i++)
  printf("\n arr[%d] = %d", i, arr[i]);
 getch();
 return 0;
}
```

**Input:**

```
Enter the number of elements in the array : 5
arr[0] = 1
arr[1] = 2
arr[2] = 3
arr[3] = 4
arr[4] = 5
Enter the position from which the number has to be deleted : 3
```

**Output:**

```
The array after deletion is :
arr[0] = 1
arr[1] = 2
arr[2] = 3
arr[3] = 5
```

**Lab assignment:**

1) Merging two arrays
2) Sorting an array in ascending or descending order

**Exercise No. 2: Implementation of linked lists: inserting, deleting a linked list.**

**Description:**

A singly linked list is the simplest type of linked list in which every node contains some data anda pointer to the next node of the same data type. By saying that the node contains a pointer to the next node, we mean that the node stores the address of the next node in sequence.

A new node is added into an already existing linked list like
Case 1: The new node is inserted at the beginning.
Case 2: The new node is inserted at the end.

Case 4: The new node is inserted before a given node.

Before we describe the algorithms to perform insertions in all these four cases, let us first discuss an important term called OVERFLOW. Overflow is a condition that occurs when AVAIL = NULL or no free memory cell is present in the system. When this condition occurs, the program must give an appropriate message.

A node is deleted from an already existing linked list like

Case 1: The first node is deleted.

Case 2: The last node is deleted.

Case 3: The node after a given node is deleted.

Before we describe the algorithms in all these three cases, let us first discuss an important termCalled UNDERFLOW. Underflow is a condition that occurs when we try to delete a node from a linked list that is empty. This happens when START = NULL or when there are no more nodes to delete.

Note that when we delete a node from a linked list, we actually have to free the memory occupied by that node. The memory is returned to the free pool so that it can be used to store other programs and data. Whatever be the case of deletion, we always change the AVAIL pointer so that it points to the address that has been recently vacated.

**Algorithm:**

Insertion(A) Inserting a Node Before a Given Node in a Linked List

```
Step 1: IF AVAIL=NULL
    Write OVERFLOWGo to Step 12
[END OF IF]
NEW_NODE
Step 2: SET = AVAIL
Step 3: SET AVAIL=AVAIL □□□NEXT
Step 4: SET NEW_NODE ->DATA=VAL
Step 5: SET PTR=START
Step 6: SET PREPTR=PTR
Step 7: Repeat Steps8and9while PTR DATA != NUM
    Step 8: SET PREPTR=PTR
    Step 9: SET PTR=PTR->NEXT
[END OF LOOP]
Step 10:PREPTR->NEXT = NEW_NODE
Step 11: SET NEW_NODE-> NEXT=PTR
Step 12: EXIT
```

Insertion(B) Inserting a Node After a Given Node in a Linked List

```
Step 1: IF AVAIL=NULL
    Write OVERFLOW Go to Step 12
[END OF IF]
Step 2: SET = AVAIL->NEW_NODE
Step 3: SET AVAIL=AVAIL->NEXT
Step 4: SET DATA=VAL->NEW_NODE
Step 5: SET PTR=START
Step 6: SET PREPTR=PTR
Step 7: Repeat Steps8and9while PREPTR->DATA!= NUM
    Step 8: SET PREPTR=PTR
    Step 9: SET PTR=PTR->NEXT
[END OF LOOP]
Step 10: PREPTR->NEXT =NEW_NODE
Step 11: SET NEW_NODE->NEXT=PTR
Step 12: EXIT
```

Deletion

```
Step 1: IF START=NULL
    Write UNDERFLOW
    Go to Step 10
[END OF IF]
Step 2: SET PTR=START
Step 3: SET PREPTR=PTR
Step 5: SET PREPTR=PTR
Step 6: SET PTR=PTR->NEXT
[END OF LOOP]
Step 7: SET TEMP=PTR
Step 8: SET PREPTR->NEXT=PTR->NEXT
Step 9: FREE TEMP
Step 10:EXIT
```

**Aim:Write a program to create a linked list and perform insertions and deletions Write functions to sort and finally delete the entire list at once.**

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#include <malloc.h>
struct node
{
 int data;
 struct node *next;
};
struct node *start = NULL;
struct node *create_ll(struct node *);
struct node *display(struct node *);
struct node *insert_beg(struct node *);
struct node *insert_end(struct node *);
struct node *insert_before(struct node *);
struct node *insert_after(struct node *);
struct node *delete_beg(struct node *);
struct node *delete_end(struct node *);
struct node *delete_node(struct node *);
struct node *delete_after(struct node *);
struct node *delete_list(struct node *);
struct node *sort_list(struct node *);
int main(int argc, char *argv[]) {
 int option;
 do
 {
  printf("\n\n *****MAIN MENU *****");
  printf("\n 1:  Create a list");
  printf("\n 2:  Display the list");
  printf("\n 3:  Add a node at the beginning");
  printf("\n 4:  Add a node at the end");
  printf("\n 5:  Add a node before a given node");
  printf("\n 6:  Add a node after a given node");
  printf("\n 7:  Delete a node from the beginning");
printf("\n 8:  Delete a node from the end");
printf("\n 9:  Delete a given node");
printf("\n 10: Delete a node after a given node");
printf("\n 11: Delete the entire list");
printf("\n 12: Sort the list");
printf("\n 13: EXIT");
printf("\n\n Enter your option : ");
  scanf("%d", &option);
  switch(option)
  {
case 1: start = create_ll(start);
 printf("\n LINKED LIST CREATED");
case 2: start = display(start);
   break;
case 3: start = insert_beg(start);
   break;
case 4: start = insert_end(start);
   break;
case 5: start = insert_before(start);
   break;
case 6: start = insert_after(start);
   break;
case 7: start = delete_beg(start);
   break;
case 8: start = delete_end(start);
   break;
case 9: start = delete_node(start);
   break;
case 10: start = delete_after(start);
   break;
case 11: start = delete_list(start);
 printf("\n LINKED LIST DELETED");
   break;
case 12: start = sort_list(start);
   break;
  }
}while(option !=13);
return 0;
struct node *create_ll(struct node *start)
struct node *new_node, *ptr;
printf("\n Enter -1 to end");
printf("\n Enter the data : ");
scanf("%d", &num);
while(num!=-1)
new_node = (struct node*)malloc(sizeof(struct node));
new_node -> data=num;
  if(start==NULL)
  {
 new_node -> next = NULL;
   start =
new_node;
  }
  else
  {
   ptr=start;
   while(ptr->next!=NULL)
   ptr=ptr->next;
   ptr->next =
new_node;
   new_node->next=NULL;
  }
  printf("\n Enter the data : ");
  scanf("%d", &num);
 }
 return start;
}
struct node *display(struct node *start)
{
 ptr = start;
 while(ptr != NULL)
 {
  printf(“\t %d”, ptr -> data);
  ptr = ptr -> next;
 }
 return start;
}
struct node *insert_beg(struct node *start)
{
 struct node *new_node;
 int num;
 printf(“\n Enter the data : “);
 scanf(“%d”, &num);
 new_node = (struct node *)malloc(sizeof(struct node));
 new_node -> data = num;
 new_node -> next = start;
 start = new_node;
 return start;
}
struct node *insert_end(struct node *start)
{
 struct node *ptr, *new_node;
 int num;
 printf(“\n Enter the data : “);
 scanf(“%d”, &num);
 new_node = (struct node *)malloc(sizeof(struct node));
 new_node -> data = num;
 new_node -> next = NULL;
 ptr = start;
 while(ptr -> next != NULL)
 ptr = ptr -> next;
 ptr -> next = new_node;
 return start;
}
struct node *insert_before(struct node *start)
{
 struct node *new_node, *ptr, *preptr;
 int num, val;
 printf(“\n Enter the data : “);
 scanf(“%d”, &num);
 printf(“\n Enter the value before which the data has to be inserted : “);
 scanf(“%d”, &val);
 new_node = (struct node *)malloc(sizeof(struct node));
 new_node -> data = num;
 ptr = start;
 while(ptr -> data != val)
 {
  preptr = ptr;
  ptr = ptr -> next;
 }
 preptr -> next = new_node;
 new_node -> next = ptr;
 return start;
}
struct node *insert_after(struct node *start)
```

```
 struct node *new_node, *ptr, *preptr;
 int num, val;
 printf(“\n Enter the data : “);
 scanf(“%d”, &num);
 printf(“\n Enter the value after which the data has to be inserted : “);
scanf(“%d”, &val);
new_node = (struct node *)malloc(sizeof(struct node));
new_node -> data = num;
ptr = start;
preptr = ptr;
while(preptr -> data != val)
 {
 preptr = ptr;
 ptr = ptr -> next;
 }
preptr -> next=new_node;
new_node -> next = ptr;
return start;
struct node *delete_beg(struct node *start)
struct node *ptr;
ptr = start;
start = start -> next;
 free(ptr);
return start;
struct node *delete_end(struct node *start)
struct node *ptr, *preptr;
ptr = start;
while(ptr -> next != NULL)
 {
 preptr = ptr;
 ptr = ptr -> next;
 }
preptr -> next = NULL;
 free(ptr);
return start;
struct node *delete_node(struct node *start)
struct node *ptr, *preptr;
int val;
printf(“\n Enter the value of the node which has to be deleted : “);
scanf(“%d”, &val);
ptr = start;
if(ptr -> data == val)
 {
start = delete_beg(start);
return start;
}
 else
 {
while(ptr -> data != val)
  {
   preptr = ptr;
   ptr = ptr -> next;
  }
  preptr -> next = ptr -> next;
  free(ptr);
  return start;
```

```
}
struct node *delete_after(struct node *start)
{
 struct node *ptr, *preptr;
 int val;
 printf(“\n Enter the value after which the node has to deleted : “);
 scanf(“%d”, &val);
 ptr = start;
 preptr = ptr;
 while(preptr -> data != val)
 {
  preptr = ptr;
  ptr = ptr -> next;
 }
 preptr -> next=ptr -> next;
 free(ptr);
 return start;
}
struct node *delete_list(struct node *start)
{
              struct
              node     *ptr;
 if(start!=NULL){
  ptr=start;
  while(ptr != NULL)
  {
   printf(“\n %d is to be deleted next”, ptr -> data);
   start =
delete_beg(ptr);
   ptr =
start;
  }
 }

 return start;
}
struct node *sort_list(struct node *start)
{
 struct node *ptr1, *ptr2;
 int temp;
 ptr1 = start;
 while(ptr1 -> next != NULL)
 {
  ptr2 = ptr1 -> next;
  while(ptr2 != NULL)
  {
   if(ptr1 -> data > ptr2 -> data)
   {
             temp = ptr1 -> data;
             ptr1 -> data = ptr2 -> data;
             ptr2 -> data = temp;
   }
   ptr2 = ptr2 -> next;
  }
  ptr1 = ptr1 -> next;
 }
```

```
}
```

**Input:**

```
3
4
5
```

**Output:**

```
*****MAIN MENU *****
1: Create a list
2: Display the list
3: Add a node at the beginning
4: Add the node at the end
5: Add the node before a given node
6: Add the node after a given node
7: Delete a node from the beginning
8: Delete a node from the end
9: Delete a given node
10: Delete a node after a given node
11: Delete the entire list
12: Sort the list
13: Exit

Enter your option : 1
Enter the data :3
Enter your option : 2
3
Enter your option : 3
Enter the data : 4
Enter your option : 6
Add after given node:4
Enter the data : 5
Enter your option : 2
4       5       3
Enter your option : 10
Delete after a given node:5
Enter your option : 2
4       5
```

**Lab Assignment:**

1) WAP to implement circular linked list.
2) WAP to insert and delete an element in a doubly linked list(all cases).

## Exercise No. 3: Stacks and Queues: adding, deleting elements

**Description:**

A stack is a linear data structure which uses the same principle, i.e., the elements in a stack are added and removed only from one end, which is called theTOP. Hence, a stack is called a LIFO (Last-In First-Out) datastructure, as the element that was inserted last is the first one to be taken out.

A stack supports three basic operations: push, pop, and peek. The push operation adds an element to the top of the stack and the pop operation removes the element from the top of the stack. The peek operation returns the value of the topmost element of the stack.

**Aim: Write a program to perform Push, Pop, and Peek operations on a stack.**

**Algorithm:**

```
Insertion:
Step 1: IF TOP=MAX-1
                PRINT OVERFLOW
        Goto Step 4
```

```
Step 2: SET TOP=TOP+1
Step 3: SET STACK[TOP]=VALUE
Step 4: END
```

Deletion:

```
Step 1: IF TOP=NULL
                PRINT UNDERFLOW
        Goto Step 4
        [END OF IF]
Step 2: SET VAL=STACK[TOP]
Step 3: SET TOP=TOP-1
Step 4: END
```

Peek:

```
Step 1: IF TOP=NULL
                PRINT STACK IS EMPTY
        Goto Step 3
Step 2: RETURN STACK[TOP]
Step 3: END
```

**Program:**

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#define MAX 3 // Altering this value changes size of stack created
int st[MAX], top=-1;
void push(int st[], int val);
int pop(int st[]);
int peek(int st[]);
void display(int st[]);
int main(int argc, char *argv[]) {
 int val, option;
 do
 {
  printf("\n *****MAIN MENU*****");
  printf("\n 1. PUSH");
  printf("\n 2. POP");
  printf("\n 3. PEEK");
  printf("\n 4. DISPLAY");
  printf("\n 5. EXIT");
  printf("\n Enter your option: ");
  scanf("%d", &option);
  switch(option)
  {
  case 1:
   printf("\n Enter the number to be pushed on stack: ");
   scanf("%d", &val);
   push(st, val);
   break;
  case 2:
   val = pop(st);
   if(val != -1)
   printf("\n The value deleted from stack is: %d", val);
   break;
  case 3:
   if(val != -1)
   printf("\n The value stored at top of stack is: %d", val);
   break;
  case 4:
   display(st);
   break;
  }
 }while(option != 5);
 return 0;
}
void push(int st[], int val)
{
 if(top == MAX-1)
 {
  printf("\n STACK OVERFLOW");
 }
 else
 {
  top++;
  st[top] = val;
 }
}
int pop(int st[])
{
 int val;
 if(top == -1)
 {
  printf("\n STACK UNDERFLOW");
  return -1;
 }
 else
 {
  val = st[top];
  top--;
  return val;
 }
}
void display(int st[])
{
 int i;
 if(top == -1)
 printf("\n STACK IS EMPTY");
 else
 {
  for(i=top;i>=0;i--)
  printf("\n %d",st[i]);
  printf("\n"); // Added for formatting purposes
 }
}
int peek(int st[])
{
 if(top == -1)
 {
  printf("\n STACK IS EMPTY");
  return -1;
 }
 return (st[top]);
}
```

**Output**

```
*****MAIN MENU*****
1. PUSH
2. POP
3. PEEK
4. DISPLAY
5. EXIT
Enter your option : 1
Enter the number to be pushed on stack : 500
Enter your option : 1
Enter the number to be pushed on stack : 700
Enter your option : 4
700        500
Enter your option : 3
Enter your option : 4
700
Enter your option : 2
Enter your option : 4
500
```

**Description:**

A queue is a FIFO (First-In, First-Out) data structure in which the element that is inserted first is the first one to be taken out. The elements in a queue are added at one end called the REAR and removed from the other end called the FRONT. Queues can be implemented by using either arrays or linked lists.

**Aim: Write a program to perform Insertion, Deletion, and Peek operations on a queue.**

**Algorithm:**

Insertion:

```
Step 1: IF REAR=MAX-1
                Write OVERFLOW
        Goto step 4
        [END OF IF]
Step 2: IF FRONT=-1 and REAR=-1
                SET FRONT=REAR =ELSE
                SET REAR=REAR+1
        [END OF IF]
Step 3: SET QUEUE[REAR]=NUM
Step 4: EXIT
```

Deletion:

```
Step 1: IF FRONT=-1OR FRONT>REAR
                Write UNDERFLOW
        ELSE
                SET VAL=QUEUE[FRONT]
                SET FRONT=FRONT+1
        [END OF IF]
Step 2: EXIT
```

**Program:**

```
#include <stdio.h>
#include <conio.h>
#define MAX 10 // Changing this value will change length of array
int queue[MaX];
int front = -1, rear = -1;
void insert(void);
int delete_element(void);
int peek(void);
void display(void);
int main()
{
 do
 {
  printf(“\n\n ***** MAIN MENU *****”);
  printf(“\n 1. Insert an element”);
  printf(“\n 2. Delete an element”);
  printf(“\n 3. Peek”);
  printf(“\n 4. Display the queue”);
  printf(“\n 5. EXIT”);
  printf(“\n Enter your option : “);
  scanf(“%d”, &option);
  switch(option)
  {
  case 1:
   insert();
   break;
  case 2:
   val = delete_element();
   if (val != -1)
   printf(“\n The number deleted is : %d”, val);
   break;
  case 3:
   val = peek();
   if (val != -1)
   printf(“\n The first value in queue is : %d”, val);
   break;
  case 4:
   display();
   break;
  }
 }while(option != 5);
 getch();
 return 0;
}
void insert()
{
 int num;
 printf(“\n Enter the number to be inserted in the queue : “);
 scanf(“%d”, &num);
 if(rear == MAX-1)
 printf(“\n OVERFLOW”);
 else if(front == -1 && rear == -1)
 front = rear = 0;
 else
 rear++;
 queue[rear] = num;
}
int delete_element()
{
 int val;
if(front == -1 || front>rear)
 {
 printf(“\n UNDERFLOW”);
 return -1;
 }
 else
 {
 val = queue[front];
 front++;
 if(front > rear)
 front = rear = -1;
 return val;
 }
```

```
if(front==-1 || front>rear)
 {
 printf("\n QUEUE IS EMPTY");
 return -1;
 }
 else
 {
 return queue[front];
 }
void display()
int i;
printf("\n");
if(front == -1 || front > rear)
printf("\n QUEUE IS EMPTY");
 else
 {
 for(i = front;i <= rear;i++)
 printf("\t %d", queue[i]);
 }
```

**Output:**

```
***** MAIN MENU *****"
1. Insert an element
2. Delete an element
3. Peek
4. Display the queue
5. Exit
Enter your option : 1
Enter the number to be inserted in the queue : 50
```

## Exercise No. 4: Evaluation Problem: Evaluation of infix to postfix expressions on stack.

**Description:**

Infix, postfix, and prefix notations are three different but equivalent notations of writing algebraic expressions. For example, if an expression is written as `A+B` in infix notation, the same expression can be written as `AB+` in postfix notation. The order of evaluation of a postfix expression is always from left to right. Even brackets cannot alter the order of evaluation. The expression `(A+B)*C` can be written as: `[AB+]*C` =>`AB+C*` in the postfix notation.

**Aim:Write a program to convert a given infix expression into its postfix Equivalent, Implement the stack using an array.**

**Algorithm:**

```
Step 1: Add)to the end of the infix expression
Step 2: Push(onto the stack
Step 3: Repeat until each character in the infix notation is scanned
IF a(is encountered, push it on the stack
IF an operand (whetheradigit oracharacter) is encountered, add it to thepostfix expression.
IF a)is encountered, then
        a. Repeatedly pop from stack and add it to the postfix expression until a
(is encountered.
        b. Discard the (.That is, remove the(from stack and do notadd it to the postfix expression
IF an operator is encountered, then
        a. Repeatedly pop from stack and add each operator (popped from the stack) to thepostfix expression
which has the same precedence orahigher precedence than )
        b. Push the operator to the stack
[END OF IF]
Step 4: Repeatedly pop from the stack and add it to the postfix expression until the stack is empty
Step 5: EXIT
```

**Program:**

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
```

```
char stack[MAX];
int top=1;
char pop();  /*declaration of  pop function*/
void push(char item); /*declaration of  push function*/
int prcd(char symbol)  /*checking the precedence*/
{
  switch(symbol) /*assigning values for symbols*/
  {
    case '+':
    case '-': return 2;
    break;
    case '*':
    case '/': return 4;
    break;
    case '^':return 6;
    break;
    case '(':
    case ')':
    case '#':return 1;
   break;
  }
}
int(isoperator(char symbol))  /*assigning  operators*/
{
  switch(symbol)
  {
     case '+':
     case '*':
  case '-':
    case '/':
    case '^':
    case '(':
    case ')':return 1;
    break;
    default:return 0;
  }
 }
/*converting infix to postfix*/
void convertip(char infix[],char postfix[])
{
 int i,symbol,j=0;
 stack[++top]='#';
 for(i=0;i<strlen(infix);i++)
 {
  symbol=infix[i];
  if(isoperator(symbol)==0)
  {
    postfix[j]=symbol;
     j++;
  }
  else
  {
    if(symbol=='(')
    push(symbol);   /*function call  for pushing elements into the stack*/
    else if(symbol==')')
    {
       {
          postfix[j]=pop();
          j++;
       }
       pop();     /*function call  for popping elements into the stack*/
    }
   else
   {
       if(prcd(symbol)>prcd(stack[top]))
       push(symbol);
       else
       {
          while(prcd(symbol)<=prcd(stack[top]))
          {
            postfix[j]=pop();
             j++;
          }
         push(symbol);
       }/*end of else loop*/
     }/*end of else loop*/
    } /*end of else loop*/
   }/*end of for loop*/
  While (stack[top]!='#')
  {
    postfix[j]=pop();
     j++;
   }
   postfix[j]='\0'; /*null terminate string*/
}
/*main program*/
void main()
 {
     char infix[20],postfix[20];
     printf("enter the valid infix string \n");
     gets(infix);
     convertip(infix,postfix);    /*function call   for converting infix to postfix */
     printf("the corresponding postfix string is:\n");
    puts(postfix);
  }
/*push operation*/
void push(char item)
{
   top++;
   stack[top]=item;
 }
/*pop operation*/
char pop()
{
   char a;
   a=stack[top];
   top--;
   return a;
 }
```

**Input:**

A+B*C

**Output:**

## Exercise No. 5: Circular Queue: Adding & deleting elements

**Description:**

In the circular queue, the first index comes right after the last index.The circular queue will be full only when `FRONT=0` and `REAR=Max-1`. A circular queue is implemented in the same manner as a linear queue is implemented.

**Aim: Write a program to implement a circular queue using array.**

**Algorithm:**

**Insertion:**

```
Step 1: IF FRONT = and Rear=MAX-1
            Write OVERFLOW
            Goto step 4
        [End OF IF]

Step 2:
IF FRONT=-1 and REAR=-1
        SET FRONT=REAR =0
ELSE IF REAR=MAX-1and FRONT !=0
            SET REAR =0
  ELSE
            SET REAR=REAR+1
[END OF IF]
Step 3: SET QUEUE[REAR]=VAL
Step 4: EXIT
```

Deletion:

```
Step 1: IF FRONT=-1
            Write UNDERFLOW
            Goto Step 4
        [END of IF]
Step 2: SET VAL=QUEUE[FRONT]
Step 3: IF FRONT=REAR
            SET FRONT=REAR=-1
        ELSE
            IF FRONT=MAX -1
                SET FRONT =0
            ELSE
                SET FRONT=FRONT+1
            [END of IF]
        [END OF IF]
Step 4: EXIT
```

**Program:**

```
#include <stdio.h>
#include <conio.h>
#define MAX 10
int queue[MAX];
int front=–1, rear=–1;
void insert(void);
int delete_element(void);
int peek(void);
void display(void);
int main()
{
 int option, val;
 clrscr();
 do
 {
  printf("\n ***** MAIN MENU *****");
  printf("\n 1. Insert an element");
  printf("\n 2. Delete an element");
  printf("\n 3. Peek");
  printf("\n 4. Display the queue");
  printf("\n Enter your option : ");
  scanf("%d", &option);
  switch(option)
  {
   case 1:
    insert();
    break;
   case 2:
    val = delete_element();
    if(val!=–1)
    printf("\n The number deleted is : %d", val);
    break;
   case 3:
    val = peek();
    if(val!=–1)
    printf("\n The first value in queue is : %d",     break;
   case 4:
    display();
    break;
  }
 }while(option!=5);
 getch();
 return 0;
}
void insert()
{
 int num;
 printf("\n Enter the number to be inserted in the queue : ");
 scanf("%d", &num);
 if(front==0 && rear==MAX–1)
  printf("\n OVERFLOW");
 else if(front==–1 && rear==–1)
 {
  front=rear=0;
  queue[rear]=num;
 }
 else if(rear==MAX–1 && front!=0)
 {
  rear=0;
  queue[rear]=num;
 }
 else
 {
  rear++;
  queue[rear]=num;
 }
}
int delete_element()
{
 int val;
 if(front==–1 && rear==–1)
  {
  printf("\n UNDERFLOW");
  return –1;
  }
 val = queue[front];
 if(front==rear)
  front=rear=–1;
 else
 {
  if(front==MAX–1)
   front=0;
   front++;
  }
 return val;
}
int peek()
{
 if(front==–1 && rear==–1)
 {
  printf("\n QUEUE IS EMPTY");
  return –1;
}
 else
 {
  return queue[front];
 }
}
void display()
{
 int i;
 printf("\n");
 if (front ==–1 && rear= =–1)
  printf ("\n QUEUE IS EMPTY");
 else
 {
  if(front<rear)
  {
   for(i=front;i<=rear;i++)
    printf("\t %d", queue[i]);
  }
  else
  {
   for(i=front;i<MAX;i++)
    printf("\t %d", queue[i]);
   for(i=0;i<=rear;i++)
    printf("\t %d", queue[i]);
  }
 }
}
```

**Output**

```
***** MAIN MENU *****
1. Insert an element
2. Delete an element
3. Peek
4. Display the queue
5. EXIT
Enter your option : 1
Enter the number to be inserted in the queue : 25
Enter your option : 2
The number deleted is : 25
Enter your option : 3
QUEUE IS EMPTY
Enter your option : 5
```

## Exercise No. 6: Implementation of Polynomial addition, Polynomial multiplicationusing linked lists.

## Description:

A polynomial is represented in the memory using a linked list. Consider a polynomial $6x^3+9x^2+7x+1$. Every individual term in a polynomial consists of two parts, a coefficientand a power. Here, 6, 9, 7, and 1 are the coefficients of the terms that have 3, 2, 1, and 0 as theirpowers respectively.

Every term of a polynomial can be represented as a node of the linked list

**Aim:Write a program to add two polynomials.**

**Program:**

```
#include <stdio.h>
typedef struct pnode
{
float coef;
int exp;
struct pnode *next;
}p;
p *getnode();
void main()
{
p *p1,*p2,*p3;

p *getpoly(),*add(p*,p*);

void display(p*);
clrscr();
printf(“\n enter first polynomial”);
p1=getpoly();
printf(“\n enter second polynomial”);
p2=getpoly();
printf(“\nthe first polynomial is”);
display(p1);
printf(“\nthe second polynomial is”);
display(p2);
p3=add(p1,p2);
printf(“\naddition of two polynomial is :\n”);
display(p3);

}
p *getpoly()
{
p *temp,*New,*last;
int flag,exp;
char ans;
float coef;
temp=NULL;
flag=1;
printf(“\nenter the polynomial in descending order of exponent”);
do
{
printf(“\nenter the coef & exponent of a term”);
scanf(“%f%d”,&coef,&exp);
New=getnode();
if(New==NULL)
printf(“\nmemory cannot be allocated”);
New->coef=coef;
New->exp=exp;
if(flag==1)
{
temp=New;
last=temp;
flag=0;
}
else
{
last->next=New;
last=New;
}
ans=getch();
}
while(ans==’y’);
return(temp);
}
p *getnode()
{
p *temp;
temp=(p*) malloc (sizeof(p));
temp->next=NULL;
return(temp);
}
void display(p*head)
{
p*temp;
temp=head;
if(temp==NULL)
printf(“\npolynomial empty”);
while(temp->next!=NULL)
{
printf(“%0.1fx^%d+”,temp->coef,temp->exp);
temp=temp->next;
}
printf(“\n%0.1fx^%d”,temp->coef,temp->exp);
getch();
}
p*add(p*first,p*second)
{
p *p1,*p2,*temp,*dummy;
char ch;
float coef;
p *append(int,float,p*);
p1=first;
p2=second;
temp=(p*)malloc(sizeof(p));
if(temp==NULL)
printf(“\nmemory cannot be allocated”);
dummy=temp;
while(p1!=NULL&&p2!=NULL)
{
if(p1->exp==p2->exp)
{
coef=p1->coef+p2->coef;
temp=append(p1->exp,coef,temp);
p1=p1->next;
p2=p2->next;
}
else
if(p1->expexp)
{
coef=p2->coef;
temp=append(p2->exp,coef,temp);
p2=p2->next;
}
else
if(p1->exp>p2->exp)
{
coef=p1->coef;
temp=append(p1->exp,coef,temp);
p1=p1->next;
}
}
```

```
{
temp=append(p1->exp,p1->coef,temp);
p1=p1->next;
}
while(p2!=NULL)
{
temp=append(p2->exp,p2->coef,temp);
p2=p2->next;
}
temp->next=NULL;
temp=dummy->next;
free(dummy);
return(temp);
}
p*append(int Exp,float Coef,p*temp)
{
p*New,*dum;
New=(p*)malloc(sizeof(p));
if(New==NULL)
printf("\ncannot be allocated");
New->exp=Exp;
New->coef=Coef;
New->next=NULL;
dum=temp;
dum->next=New;
dum=New;
return(dum);
}
```

**Input:**

```
A^2+2A+2
A^3+3A+3
```

**Output:**

```
A^3+A^2+5A+5
```

**Lab Assignment:**

1) Write a program to multiply two polynomials.

## Exercise No. 7: Sparse Matrices: Multiplication, addition.

**Description:**

Sparse matrix is a matrix that has large number of elements with a zero value. In order to efficiently utilize the memory, specialized algorithms and data structures that take advantage of the sparse structure should be used. If we apply the operations using standard matrix structures and algorithms to sparse matrices, then the execution will slow down and the matrix will consume large amount of memory. Sparse data can be easily compressed, which in turn can significantly reduce memory usage.

**Aim: Write a program to multiply sparse matrices.**

**Program:**

```
#include<stdio.h>
#include<conio.h>
#include<alloc.h>
#define MAX1 3
#define MAX2 3
#define MAXSIZE 20
#define TRUE 1
#define FALSE 2
struct sparse
{
int *sp ;
int row ;
} ;
void initsparse ( struct sparse * ) ;
void create_array ( struct sparse * ) ;
int count ( struct sparse ) ;
void display ( struct sparse ) ;
void create_tuple ( struct sparse*, struct sparse ) ;
void display_tuple ( struct sparse ) ;
void prodmat ( struct sparse *, struct sparse, struct sparse ) ;
void searchina ( int *sp, int ii, int*p, int*flag ) ;
void searchinb ( int *sp, int jj, int colofa, int*p, int*flag ) ;
void display_result ( struct sparse ) ;
void delsparse ( struct sparse * ) ;
void main( )
{
struct sparse s[5] ;
int i ;
clrscr( ) ;
for ( i = 0 ; i<= 3 ; i++ )
initsparse ( &s[i] ) ;
create_array ( &s[0] ) ;
create_tuple ( &s[1], s[0] ) ;
display_tuple ( s[1] ) ;
create_array ( &s[2] ) ;
create_tuple ( &s[3], s[2] ) ;
display_tuple ( s[3] ) ;
prodmat ( &s[4], s[1], s[3] ) ;
printf ( "\nResult of multiplication of two matrices: " ) ;
display_result ( s[4] ) ;
for ( i = 0 ; i<= 3 ; i++ )
delsparse ( &s[i] ) ;
getch( ) ;
}
/* initialises elements of structure */
void initsparse ( struct sparse *p )
{
p -> sp = NULL ;
p -> result = NULL ;
}
/* dynamically creates the matrix */
void create_array ( struct sparse *p )
{
int n, i ;
/* allocate memory */
p -> sp = ( int * ) malloc ( MAX1 * MAX2 * sizeof ( int ) ) ;
/* add elements to the array */
for ( i = 0 ; i< MAX1 * MAX2 ; i++ )
{
printf ( "Enter element no. %d: ", i ) ;
scanf ( "%d", &n ) ;
* ( p -> sp + i ) = n ;
}
}
/* displays the contents of the matrix */
```

```
{
int i ;
/* traverses the entire matrix */
for ( i = 0 ; i< MAX1 * MAX2 ; i++ )
{
/* positions the cursor to the new line for every new row */
if ( i % 3 == 0 )
printf ( "\n" ) ;
printf ( "%d\t", * ( s.sp + i ) ) ;
}
}
/* counts the number of non-zero elements */
int count ( struct sparse s )
{
int cnt = 0, i ;
for ( i = 0 ; i< MAX1 * MAX2 ; i++ )
{
if ( * ( s.sp + i ) != 0 )
cnt++ ;
}
return cnt ;
}
/* creates an array that stores information about non-zero elements */
void create_tuple ( struct sparse *p, struct sparse s )
{
int r = 0 , c = -1, l = -1, i ;
/* get the total number of non-zero elements */
p -> row = count ( s ) + 1 ;
/* allocate memory */
p -> sp = ( int * ) malloc ( p -> row * 3 * sizeof ( int ) ) ;
/* store information about total no. of rows, cols, and non-zero values */
* ( p -> sp + 0 ) = MAX1 ;
* ( p -> sp + 1 ) = MAX2 ;
* ( p -> sp + 2 ) = p -> row - 1 ;
l = 2 ;
/* scan the array and store info. about non-zero values in the 3-tuple */
for ( i = 0 ; i< MAX1 * MAX2 ; i++ )
{
c++ ;
/* sets the row and column values */
if ( ( ( i % 3 ) == 0 ) && ( i != 0 ) )
{
r++ ;
c = 0 ;
}
/* checks for non-zero element, row, column and non-zero value is assigned to the matrix */
if ( * ( s.sp + i ) != 0 )
{
l++ ;
* ( p -> sp + l ) = r ;
l++ ;
* ( p -> sp + l ) = c ;
l++ ;
```

```
}
}
}
/* displays the contents of the matrix */
void display_tuple ( struct sparse s )
{
int i, j ;
/* traverses the entire matrix */
printf ( "\nElements in a 3-tuple: " ) ;
j = ( * ( s.sp + 2 ) * 3 ) + 3 ;
for ( i = 0 ; i< j ; i++ )
{
/* positions the cursor to the new line for every new row */
if ( i % 3 == 0 )
printf ( "\n" ) ;
printf ( "%d\t", * ( s.sp + i ) ) ;
}
printf ( "\n" ) ;
}
/* performs multiplication of sparse matrices */
void prodmat ( struct sparse *p, struct sparse a, struct sparse b )
{
int sum, k, position, posi, flaga, flagb, i , j ;
k = 1 ;
p -> result = ( int * ) malloc ( MAXSIZE * 3 * sizeof ( int ) ) ;
for ( i = 0 ; i< * ( a.sp + 0 * 3 + 0 ) ; i++ )
{
for ( j = 0 ; j< * ( b.sp + 0 * 3 + 1 ) ; j++ )
{
/* search if an element present at ith row */
searchina ( a.sp, i, &position, &flaga ) ;
if ( flaga == TRUE )
{
sum = 0 ;
/* run loop till there are element at ith row in first 3-tuple */
while ( * ( a.sp + position * 3 + 0 ) == i )
{
/* search if an element present at ith col. in second 3-tuple */
searchinb ( b.sp, j, * ( a.sp + position * 3 + 1 ), &posi, &flagb ) ;
/* if found then multiply */
if ( flagb == TRUE )
sum = sum + * ( a.sp + position * 3 + 2 ) * * ( b.sp + posi * 3 + 2 ) ;
position = position + 1 ;
}
/* add result */
if ( sum != 0 )
{
* ( p -> result + k * 3 + 0 ) = i ;
* ( p -> result + k * 3 + 1 ) = j ;
* ( p -> result + k * 3 + 2 ) = sum ;
k = k + 1 ;
}
}
```

```
}
/* add total no. of rows, cols and non-zero values */
* ( p -> result + 0 * 3 + 0 ) = * ( a.sp + 0 * 3 + 0 ) ;
* ( p -> result + 0 * 3 + 1 ) = * ( b.sp + 0 * 3 + 1 ) ;
* ( p -> result + 0 * 3 + 2 ) = k - 1 ;
}
/* searches if an element present at iith row */
void searchina ( int *sp, int ii, int *p, int *flag )
{
int j ;
*flag = FALSE ;
for ( j = 1 ; j<= * ( sp + 0 * 3 + 2 ) ; j++ )
{
if ( * ( sp + j * 3 + 0 ) == ii )
{
*p = j ;
*flag = TRUE ;
return ;
}
}
}
/* searches if an element where col. of first 3-tuple is equal to row of second 3-tuple */
void searchinb ( int *sp, int jj, int colofa, int *p, int *flag )
{
int j ;
*flag = FALSE ;
for ( j = 1 ; j<= * ( sp + 0 * 3 + 2 ) ; j++ )
{
if ( * ( sp + j * 3 + 1 ) == jj && * ( sp + j * 3 + 0 ) == colofa )
{
*p = j ;
*flag = TRUE ;
return ;
}
}
}
/* displays the contents of the matrix */
void display_result ( struct sparse s )
{
int i ;
/* traverses the entire matrix */
for ( i = 0 ; i< ( * ( s.result + 0 + 2 ) + 1 ) * 3 ; i++ )
{
/* positions the cursor to the new line for every new row */
if ( i % 3 == 0 )
printf ( "\n" ) ;
printf ( "%d\t", * ( s.result + i ) ) ;
}
}
/* deallocates memory */
void delsparse ( struct sparse *s )
{
if ( s -> sp != NULL )
if ( s -> result != NULL )
free ( s -> result ) ;
}
```

**Input:**

First matrices

```
[0 2 3]
[4 0 0]
[0 0 5]
```

Second matrices

```
[0 0 7]
[0 8 0]
[0 9 6]
```

**Output:**

```
[0 43 18]
[0  0 28]
[0 45 30]
```

**Lab assignment:**

1) Write a program to add two sparse matrices.

## Exercise No. 8: Recursive and Non-recursive traversal of Trees

## Description:

A binary tree is a data structure that is defined as a collection of elements called nodes. In a binary tree, the topmost element is called the root node, and each node has 0, 1, or at the most 2 children. A node that has zero children is called a leaf node or a terminal node. Every node contains a data element, a left pointer which points to the left child, and a right pointer which points to the right child. The root element is pointed by a`'root'` pointer. If `root = NULL`, then it means the tree is empty.

**Aim: Write a program to implement a binary tree using recursion.**

**Program:**

```
#include<stdio.h>
#include<conio.h>
#include<alloc.h>
struct node
{
int data;
struct node *left,*right;
};
struct node *root;
void insert(int x)
{
   struct node *p,*previous,*current;
   p=(struct node *)malloc(sizeof(struct node));
   if(p==NULL)
   {
     printf("\n Out of memory");
   }
   p->data=x;
   p->left=NULL;
   p->right=NULL;
   if(root=NULL)
   {
     root=p;
     return;
   }
   previous=NULL;
   current=root;
   while(current!=NULL)
   {
       previous=current;
       if(p->data<current->data)
           current=current->left
```

```
            current=current->right;
            }
     if(p->data<previous->data)
          previous->left=p;
     else
 previous->right=p;
}
void inorder(struct node *t)
{
  if (t!=NULL)
  {
  inorder(t->left);
  printf("\n %5d",t->data);
  inorder (t->right);
  }
}
void del(int x)
{
   int tright=0,tleft=0;
   struct node *ptr=root;
   struct node *parent=root;
   struct node *t1=root;
   struct node *temp=root;
   while(ptr!=NULL&& ptr->data!=x)
   {
      parent=ptr;
      if (x<ptr->data)
            ptr=ptr->left;
      else
           ptr=ptr->right;
   }
   if (ptr==NULL)
   {
     printf("\n Delete element not found");
     return ;
   }
   else if(t1->data==x && (t1->left ==NULL || t1->right==NULL))
          if(t1->left==NULL)
             t1=t1->right;
          else
             t1=t1->left;
   else if (ptr->left==NULL)
     if (x<parent->data)
       parent->left=ptr->right;
     else
       parent->right=ptr->right;
   else if (ptr->right==NULL)
     if (x<parent->data)
       parent->left=ptr->left;
     else
       parent->right=ptr->left;
   else
   {
   temp=ptr;
   parent=ptr;
   if((ptr->left)>=(ptr->right))
   {
      ptr=ptr->left;
      while(ptr->right!=NULL)
      {
           tright=1;
           parent=ptr;
```

```
    }
    temp->data=ptr->data;
    if(tright)
       parent->right=ptr->left;
    else
       parent->left=ptr->left;
   }
 else
   {
   ptr=ptr->right;
   while (ptr->left!=NULL)
   {
       tleft=1;
       parent=ptr;
       ptr=ptr->left;
    }
    temp->data=ptr->data;
    if(tleft)
       parent->left=ptr->right;
    else
       parent->right=ptr->right;
   }
   free(ptr);
 }
}

void main()
{
int op,n,srchno;
root=(struct node *)malloc(sizeof(struct node));
root->data=30;
root->right=root->left=NULL;
clrscr();
do
{
   printf("\n 1.Insertion");
   printf("\n 2.Deletion");
   printf("\n 3.Inorder");
   printf("\n 4.Quit");
   printf("\n Enter your choice\n");
   scanf("%d",&op);

 switch (op)
  {
   case 1: printf("\n Enter the element to insert\n");
         scanf("%d",&n);
         insert(n);
         break;
   case 2: printf("\n Enter the element to be deleted\n");
         scanf("%d",&srchno);
         del(srchno);
         break;
   case 3: printf("\n The inorder elements are\n");
         inorder(root);
         getch();
         break;
   default: exit(0);
  }
 }while(op<4);
 getch();

 }
```

1 2 3

**Output:**

```
Enter the element to insert1
Enter the element to insert2
Enter the element to insert3
The inorder elements are
2 1 3
```

**Lab assignment:**

1) Write a program to implement a binary tree without using recursion

## Exercise No. 9: AVL tree implementation

**Description:**

An AVL tree is the same as that of a binary search tree but with a little difference.
In its structure, it stores an additional variable called theBalance Factor. Thus, every node has a balance factor associated with it. The balance factor of a node is calculated by subtracting the height of its right sub-tree from the height of its left sub-tree. A binary search tree in which every node has a balance factor of –1, 0, or 1 is said to be height balanced. A node with any other balance factor is considered to be unbalanced and requires rebalancing of the tree.

Balance factor = Height (left sub-tree) – Height (right sub-tree)

**Aim: Write a program to implement AVL tree**

**Program:**

```
#include <stdio.h>
typedef enum { FALSE ,TRUE } bool;
struct node
{
 int val;
 int balance;
 struct node *left_child;
 struct node *right_child;
};
struct node* search(struct node *ptr, int data)
{
 if(ptr!=NULL)
   if(data < ptr -> val)
   ptr = search(ptr -> left_child,data);
   else if( data > ptr -> val)
   ptr = search(ptr -> right_child, data);
 return(ptr);
}
struct node *insert (int data, struct node *ptr, int *ht_inc)
{
 struct node *aptr;
struct node *bptr;
if(ptr==NULL)
 {
 ptr = (struct node *) malloc(sizeof(struct node));
 ptr -> val = data;
 ptr -> left_child = NULL;
 ptr -> right_child = NULL;
 ptr -> balance = 0;
 *ht_inc = TRUE;
 return (ptr);
 }
if(data < ptr -> val)
 {
 ptr -> left_child = insert(data, ptr -> left_child, ht_inc);
 if(*ht_inc==TRUE)
  {
   switch(ptr -> balance)
   {
```

```
case -1: /* Right heavy */


ptr -> balance = 0;
   *ht_inc = FALSE;
   break;


case 0: /* Balanced */


ptr -> balance = 1;


break;
   case 1: /* Left heavy */


aptr = ptr -> left_child;
    if(aptr -> balance == 1)
     {


printf(“Left to Left Rotation\n”);


ptr -> left_child= aptr -> right_child;


aptr -> right_child = ptr;


ptr -> balance = 0;


aptr -> balance=0;
     ptr = aptr;
     }
    else
     {
     printf(“Left to right rotation\n”);


bptr = aptr -> right_child;
     aptr -> right_child = bptr -> left_child;


bptr -> left_child = aptr;


ptr -> left_child = bptr -> right_child;


bptr -> right_child = ptr;


if(bptr -> balance == 1 )


pt
r -> balance = -1;
```

```
else


pt
r -> balance = 0;


if(bptr -> balance == -1)


aptr -> balance = 1;


else


aptr -> balance = 0;


bptr -> balance=0;


ptr = bptr;
     }
    *ht_inc = FALSE;
    }
  }
 }
if(data > ptr -> val)
{
  ptr -> right_child = insert(info, ptr -> right_child, ht_inc);
  if(*ht_inc==TRUE)
  {
   switch(ptr -> balance)
   {


case 1: /* Left heavy */
     ptr -> balance = 0;
     *ht_inc = FALSE;
     break;


case 0: /* Balanced */
     ptr -> balance = -1;
     break;
    case -1: /* Right heavy */

 aptr = ptr -> right_child;
  if(aptr -> balance == -1)
     {
    printf(“Right to Right Rotation\n”);
  ptr -> right_child= aptr -> left_child;
    aptr -> left_child = ptr;
      ptr -> balance = 0;
      aptr -> balance=0;
  ptr = aptr;
     }
  else
     {
  printf(“Right to Left Rotation\n”);
  aptr -> left_child = bptr -> right_child;
bptr -> right_child = aptr;
ptr -> right_child = bptr -> left_child;
  bptr -> left_child = pptr;
if(bptr -> balance == -1)
ptr -> balance = 1;
      else
ptr -> balance = 0;
      if(bptr -> balance == 1)
  aptr -> balance = -1;
      else
  aptr -> balance = 0;
      bptr -> balance=0;
  ptr = bptr;
}/*End of else*/
*ht_inc = FALSE;
   }
  }
 }
 return(ptr);
}
void display(struct node *ptr, int level)
{
 int i;
 if ( ptr!=NULL )
 {
  display(ptr -> right_child, level+1);
  printf("\n");
  for (i = 0; i < level; i++)
   printf(" ");
  printf("%d", ptr -> val);
  display(ptr -> left_child, level+1);
 }
}
void inorder(struct node *ptr)
{
 if(ptr!=NULL)
 {
  inorder(ptr -> left_child);
  printf("%d ",ptr -> val);
  inorder(ptr -> right_child);
 }
}
main()
{
 bool ht_inc;
 int data ;
 int option;
 struct node *root = (struct node *)malloc(sizeof(struct node));
root = NULL;
while(1)
 {
 printf("1.Insert\n");
 printf("2.Display\n");
 printf("3.Quit\n");
 printf("Enter your option : ");
 scanf("%d",&option);
 switch(choice)
   {
   case 1:
   printf("Enter the value to be inserted : ");
   scanf("%d", &data);
```

```
    root = insert(data, root, &ht_inc);
 else
 printf("Duplicate value ignored\n");
  break;
 case 2:
   if(root==NULL)
    {
 printf("Tree is empty\n");
    continue;
    }
   printf("Tree is :\n");
   display(root, 1);
 printf("\n\n");
printf("Inorder Traversal is: ");
 inorder(root);
   printf("\n");
   break;
  case 3:
   exit(1);
 default:
 printf("Wrong option\n");
    }
  }
 }
```

**Input:**

```
6    11    2    4    3    5
```

**Output:**

```
2    3    5    4    6    11
```

**Lab Assignment:**

1) Write a program to implement AVL tree

## Exercise No. 10: Application of sorting and searching algorithms

**Description:**

To search an element in an array is known as searching and to sort the element in an ascending and descending order is known as sorting.Two type of searching linear and binary. Mainly five type of sorting like bubble ,insertion ,selection, merge and quick sort.here we mainly focus on binary search and merge and quick sort.

**Aim:Implement Binary search without using recursion**

**Program:**

```
#include<stdio.h>

int main(){

    int a[10],i,n,m,c=0,l,u,mid;

    printf("Enter the size of an array: ");
    scanf("%d",&n);

    printf("Enter the elements in ascending order: ");
    for(i=0;i<n;i++){
         scanf("%d",&a[i]);
    }

    printf("Enter the number to be search: ");
    scanf("%d",&m);

    l=0,u=n-1;
    while(l<=u){
         mid=(l+u)/2;
```

```
            c=1;
            break;
        }
        else if(m<a[mid]){
            u=mid-1;
        }
        else
            l=mid+1;
    }
    if(c==0)
        printf("The number is not found.");
    else
        printf("The number is found.");

    return 0;
}
```

**OUTPUT:**

```
Enter the size of an array: 5
        Enter the element in ascending order: 2  4  8  9  12
        Enter the number to be  search: 3
        The number is not found.
```

### Aim: Implement Merge Sort using Divide and Conquer approach

**Program:**

```
#include<stdio.h>
#include<conio.h>
void merge(int [],int ,int ,int );
void part(int [],int ,int );
int main()
{
 int arr[30];
 int i,size;
 printf("\n\t------- Merge sorting method -------\n\n");
 printf("Enter total no. of elements : ");
 scanf("%d",&size);
 for(i=0; i<size; i++)
 {
   printf("Enter %d element : ",i+1);
   scanf("%d",&arr[i]);
 }
 part(arr,0,size-1);
 printf("\n\t------- Merge sorted elements -------\n\n");
 for(i=0; i<size; i++)
 printf("%d ",arr[i]);
 getch();
 return 0;
}

void part(int arr[],int min,int max)
{
 int mid;
 if(min<max)
```

```
   mid=(min+max)/2;
   part(arr,min,mid);
   part(arr,mid+1,max);
   merge(arr,min,mid,max);
 }
}

void merge(int arr[],int min,int mid,int max)
{
  int tmp[30];
  int i,j,k,m;
  j=min;
  m=mid+1;
  for(i=min; j<=mid && m<=max ; i++)
  {
     if(arr[j]<=arr[m])
     {
         tmp[i]=arr[j];
         j++;
     }
     else
     {
         tmp[i]=arr[m];
         m++;
     }
  }
  if(j>mid)
  {
     for(k=m; k<=max; k++)
     {
         tmp[i]=arr[k];
         i++;
     }
  }
  else
  {
     for(k=j; k<=mid; k++)
     {
        tmp[i]=arr[k];
        i++;
     }
  }
  for(k=min; k<=max; k++)
     arr[k]=tmp[k];
}
```

**Output:**

```
Enter the no of elements:7
7 8 9 4 5 3 1
The unsorted list is: 7 8 9 4 5 3 1
```

1 3 4 5 7 8 9

**Aim:Implement Quick Sort using Divide and Conquer approach**

**Program:**

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#define MAX 6000

void quick(int x[],int lb,int ub);
int partition(int x[],int lb,int ub);

void main()
{
        int i,n,x[MAX];
        time_t start,end;
        clrscr();
        printf("Enter the number of elements: ");
        scanf("%d",&n);

        for(i=0;i<n;i++)
                x[i]=rand();

        printf("\nEntered array is \n");
        for(i=0;i<n;i++)
                printf("%d ",x[i]);

        start=time(NULL);
        quick(x,0,n-1);
        end=time(NULL);
        printf("Sorted array is as shown:\n");
        for(i=0;i<n;i++)
                printf("%d ",x[i]);
        printf("\nTIME for %d elements : %f", n, difftime(end,start));
        getch();

}

void quick(int x[],int lb,int ub)
{
        int j;
        if(lb<ub)
        {
                printf("\n");
                j=partition(x,lb,ub);
                quick(x,lb,j-1);
                quick(x,j+1,ub);
        }
}

int partition(int x[],int lb,int ub)
```

```
{
        int a,down,up,temp;
        a=x[lb];
        up=ub;
        down=lb;
        while(down<up)
        {
                while(x[down]<=a&&down<ub)
                        down++;
                while(x[up]>a)
                        up--;
                if(down<up)
                {
                        temp=x[down];
                        x[down]=x[up];
                        x[up]=temp;
                }
        }
        x[lb]=x[up];
        x[up]=a;
        return up;
}
```

**Output:**

```
Enter the number of elements:5
    Entered array is
      41   18467     6334     26500     19169
    Sorted array is as shown
      41   6334       18467     19169     26500
```

# UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR

## Lab Manual

**Title of Course: Workshop Practice-II Lab**
**Course Code: ME392**
**L-T-P scheme: 0-0-3** **Course Credit: 2**

**Objectives:**

1. The objective of the laboratory is learning. The experiments are designed to illustrate phenomena in different areas of Workshop and to expose you to uses of instruments.
2. To provide an understanding of the design aspects of machines.
3. To provide an efficient understanding of the equipments and their functioning.

**Learning Outcomes:** The students will have a detailed knowledge of the concepts of process of workshop equipments and their use in various areas of mechanical engineering. Upon the completion of practical course, the student will be able to:

- **Understand** and implement basic services and functionalities of the machines using tools and equipments.
- **Use** modern manufacturing technology to understand outlined process of production.
- **Understand** the benefits of newly manufactured parts and designs.
- **Analyze** the dimensions of job and measurements to be taken in account.
- **Implement** the manufacturing processes in competition of different jobs.
- **Understand** the concepts of all different operations conducted on lathe.

**Course Contents:**
**Exercises that must be done in this course are listed below:**
Exercise No.1: Threading, Drilling, Taper Turning
Exercise No. 2: Turning, Knurling, Chamfering
Exercise No. 3: Taper Turning by Tailstock Offset Method
Exercise No. 4: Cutting Metric Thread
Exercise No. 5: Prepare Mould and cast it in Aluminum

Experiment No. 1
To perform square threading, drilling and taper turning by compound rest as per drawing
Experiment No. 2

TO PERFORM STEP TURNING, KNURLING, AND CHAMFERING ON LATHE AS PER DRAWING
Experiment No. 3

Taper turning by tailstock offset method as per drawing.
Experiment No. 4

TO CUT METRIC THREAD AS PER DRAWING
Experiment No. 5

TO PREPARE MOULD OF GIVEN PATTERN REQUIRING CORE AND TO CAST IT IN ALUMINIUM

**Text Book:**
1. Hazra Choudhary, Media Promoters & Publishers Pvt Ltd.
2. Ashish Dutt Sharma, S. Chand

## EXPERIMENT NO.- 1

### AIM

To perform square threading, drilling and taper turning by compound rest as per drawing.

### APPARATUS REQUIRED

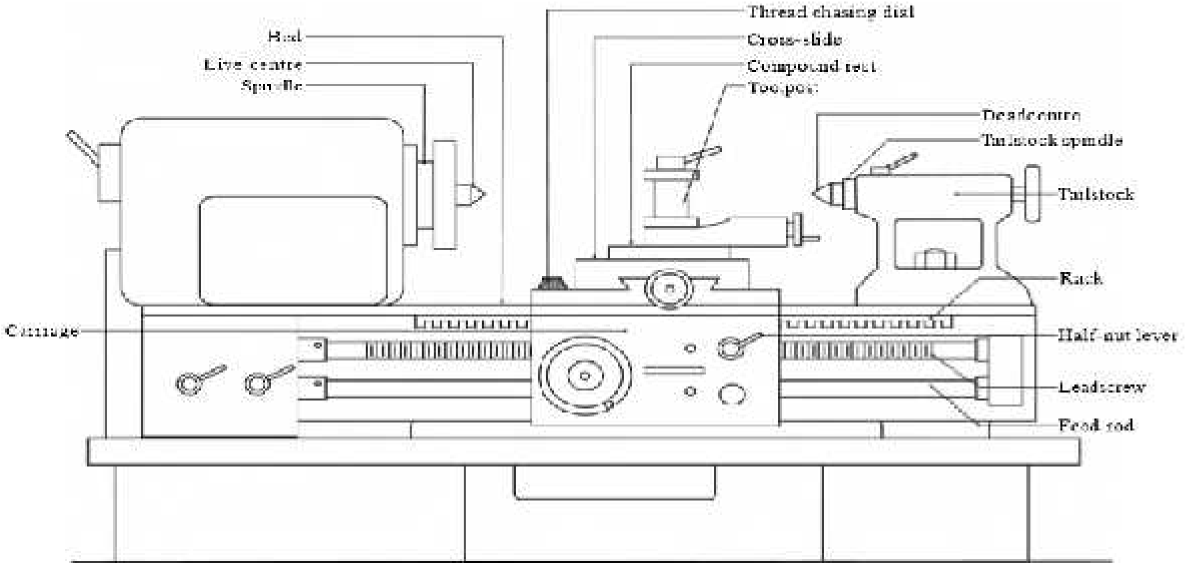
Lathe Machine with lathe tools, steel rule, outside caliper, rod of length 85x35 mm diameter.

### THEORY

INTRODUCTION

Lathe machine is a machine which performs material removal operation with tools, to produce desired shape and size of the workpiece known as machine tools. The purpose of machine tools is to save time, cost of production and to get better output which can't be obtained with hand tools.

LATHE



Lathe - Parts

The lathe is one of the most important machines in any workshop. Its main objective is to remove material by rotating the workpiece against a stationary tool. Though a lathe machine is used to produce cylindrical work, it may also be used for many other purposes such as drilling, threading, milling, grinding etc.

TYPES OF LATHE

- Speed Lathe
- Engine Lathe
- Tool Room Lathe
- Turret Lathe
- Automatic Lathe
- Automatic Bar Lathe
- Special Purpose Lathe

WORKING PRINCIPLE

In a lathe, the workpiece is held in a chuck or between centres and rotated about it's axis at a uniform speed. The cutting tool held in the tool post is fed into the workpiece for a desired depth and a desired direction [i.e. in the linear, transverse or lateral direction]. Since there exists a relationship between the workpiece and the cutting tool, the metal is removed in the form of chips and the desired shape is obtained.

PARTS OF A LATHE MACHINE

**Bed**: Usually made of cast iron, it provides a heavy, rigid frame on which all the main components are mounted.

**Ways**: Inner and outer guide rails that are precisely machined parallel to assure the accuracy of movement.

**Head Stock**: Mounted in a fixed position on the inner ways, usually at the left end using a chuck, it rotates the job.

**Gear Box**: It is fitted inside the head stock, providing multiple speeds with a geometric ratio by moving levers.

**Spindle**: Hole through the head stock by which bar stock can be fed, which allows shafts that are up to 2 times the length between Lathe centres to be worked on one end at a time.

**Chuck**: It is used to damp the job. There are basically 2 types of chucks- three jaw chuck [self-centering] and four jaw chuck [independent].

**Tail Stock**: Fits on the inner ways of the bed and can slide towards any position the headstock to fit the length of the workpiece.

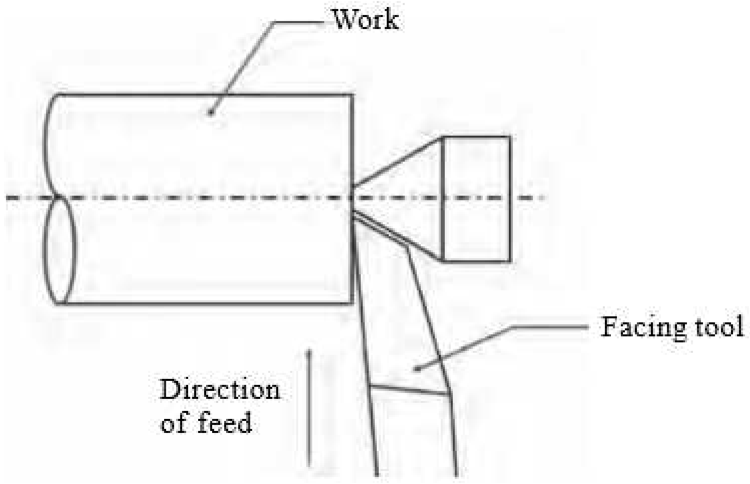
**Carriage**: Moves on the outer ways. Used for mounting and moving the cutting tools.

**Cross Slide**: Mounted on the transverse slide of the lathe and uses a hand-wheel to feed tools into the workpiece.

**Lead Screw**: For cutting threads.

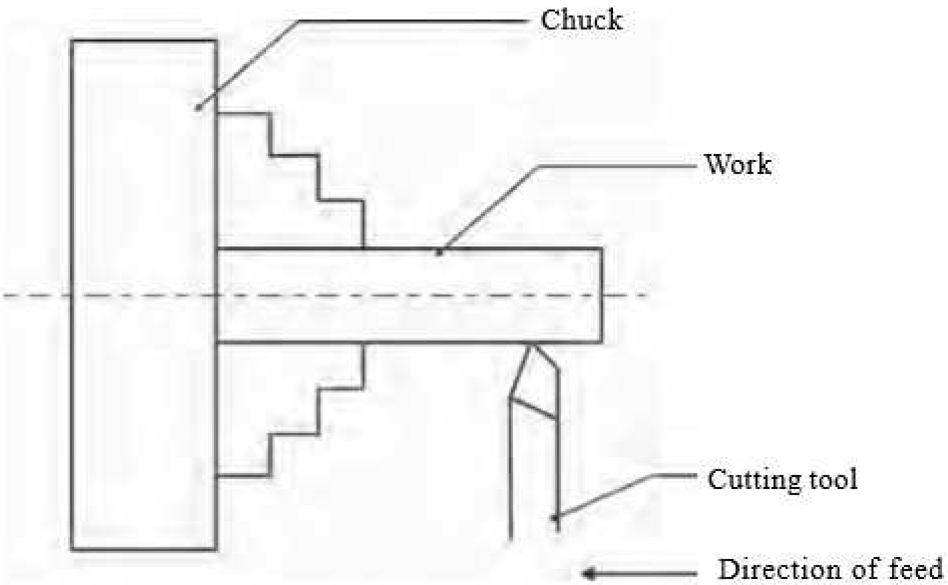
OPERATION OF A LATHE

**Facing**: The tool is fed radially into the rotating workpiece on one end to create a flat surface on the end.



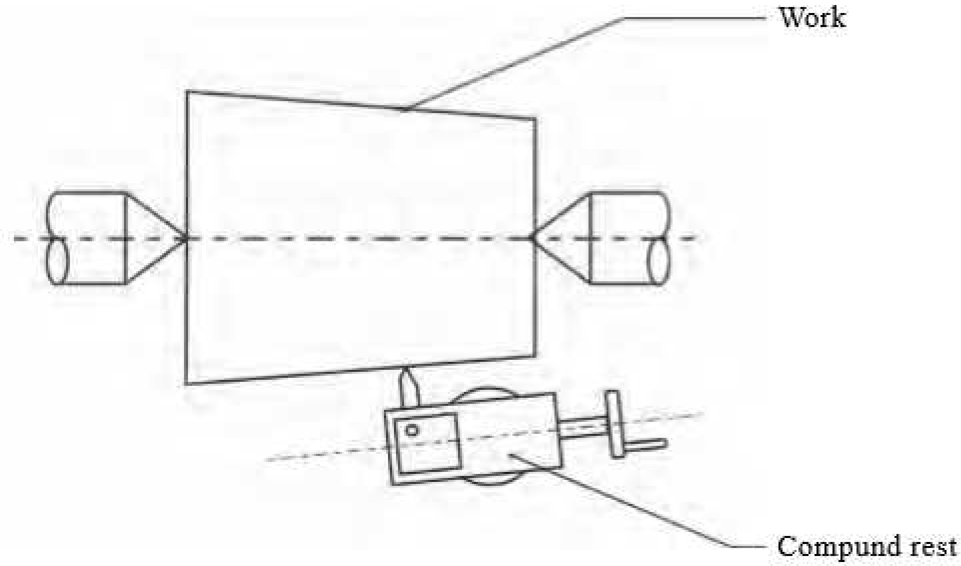
**Facing**

**Turning**: The tool is fed axially into the rotating workpiece and is used to reduce the diameter of the workpiece accordingly.



**Turning**

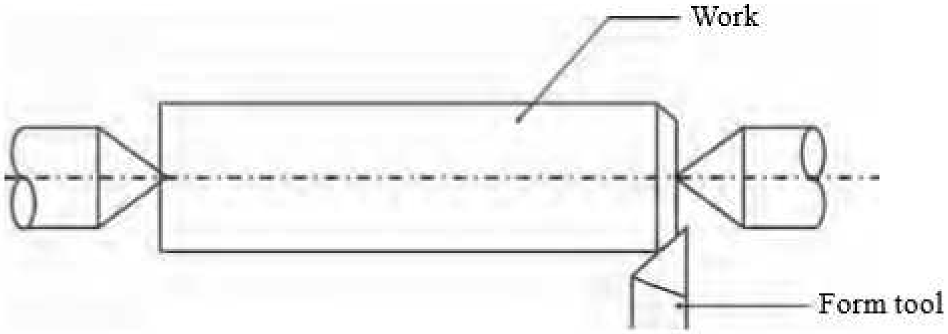
**Taper Turning**: Instead of feeding the tool parallel to the axis of rotation of the workpiece, the tool is fed at an angle, thus creating a conical geometry.



**Taper turning by compound rest method**

Contour Turning: Instead of feeding the tool along a straight line parallel to the axis of rotation as in turning, the tool follows a contour that is other than straight, thus creating a contoured form in the turned part.

**Chamfering**: The cutting edge of the tool is used to cut an angle on the corner of the cylinder forming what is called a ‘chamfer’.



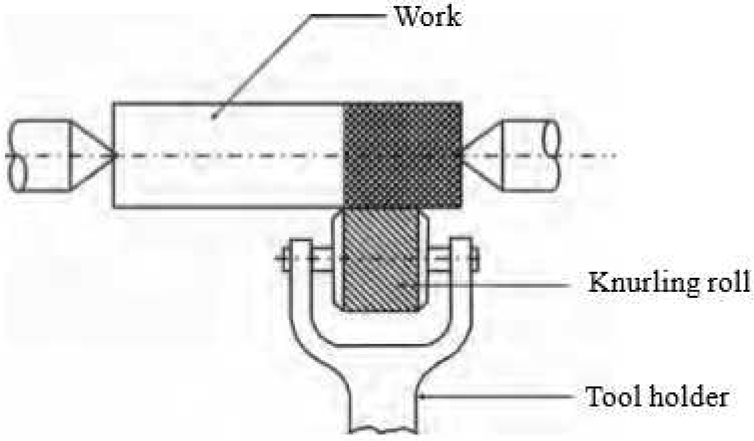
**Chamfering**

**Cut-off**: The tool is fed radially into the rotating work at some location along its length to cut off the end of the part. This operation is also known as parting.

**Threading**: A pointed tool is fed linearly across the outside surface of the rotating work part in a direction parallel to the axis of rotation at a large effective feed rate, thus creating threads over the cylinder.

**Drilling**: Drilling can be performed on a lathe by feeding the drill into the rotating work along its axis.

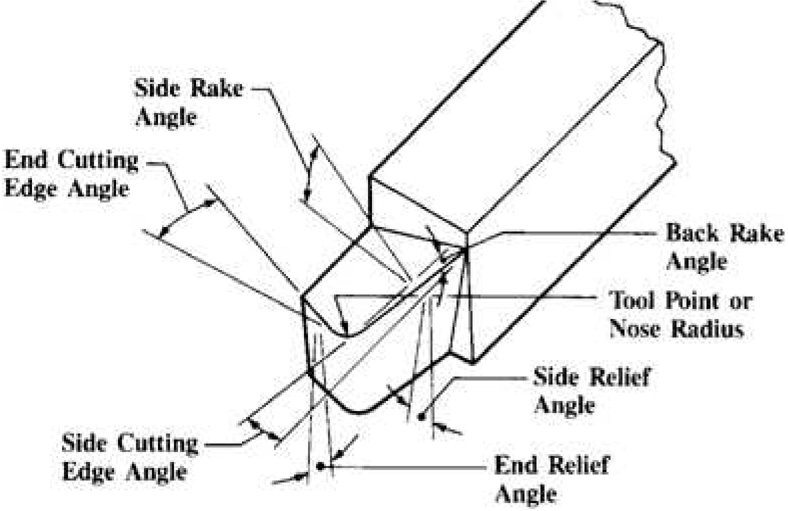
**Knurling**: This is not a machine operation because it does not involve cutting of material. Instead, it is a metal forming operation used to produce a cross-hatched pattern on the work surface.



**Knurling**

FEED MECHANISM

The feed mechanism at a lathe is employed for various feeds to the cutting tools. The feed i.e. movement of the tool relative to the workpiece may be longitudinal, cross and angular.
The feed is said to be longitudinal when the tool moves parallel to the workpiece. It is obtained by the movement of the carriage which can be repeated by hand or power.
The feed is said to be cross when the tool moves perpendicular to the workpiece. It is obtained with the help of a cross-slide which can be operated by hand or power.
The feed is said to be angular when the tool moves at an angle to the workpiece. It is obtained by moving the carriage at the appropriate angle.



## PROCEDURE

- Collect tools and material.
- Hold the job into self-centering chuck up to 20 mm.
- Set single-point cutting tool in the tool post at 45°.
- Select and set machine for a proper speed, feed for operation.
- For straight turning operation, set tool at straight to your position, lock it and make tool touch the work piece.
- Give depth of division on the cross-wheel.
- Start the machine.
- With the help of the longitudinal wheel give feed for the cutting operation till the required diameter is achieved.
- Set single-point cutting tool for chamfering at 45° to the job.
- Unclamp workpiece from the chuck.
- Hold the workpiece from backside in chuck.
- Select the tool and set for proper operation of step turning.
- Check dimensions as per the drawing and finish the job.

## RESULT

Step turning and chamfering operation performed and workpiece obtained.

## CONCLUSION

Step turning and chamfering operation done as per drawing.

## PRECAUTIONS

- Chuck key should not be left in chuck after tightening is done.
- Keep yourself away from the moving parts.
- Give the cut to workpiece carefully.
- Do not leave the machine in running condition.

## EXPERIMENT- 2

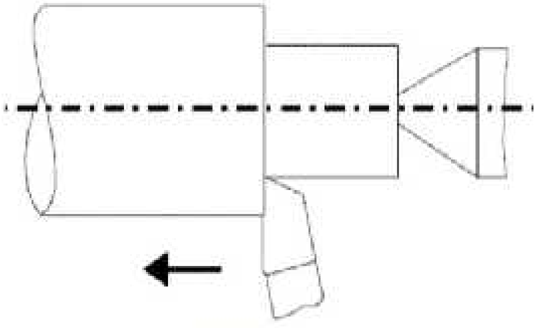
**AIM :-** TO PERFORM STEP TURNING, KNURLING, AND CHAMFERING ON LATHE AS PER DRAWING

**TOOLS AND EQUIPMENT**:-

1. CUTTING TOOL
2. LATHE MACHINE
3. VERNIER CALIPER

**THEORY:**

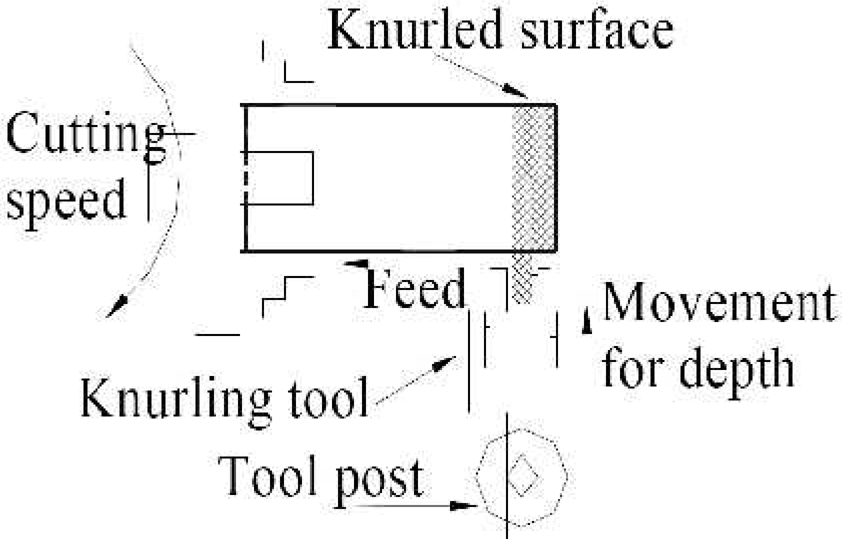
**Step Turning**: It is an operation of producing various steps of different diameters of in the work piece as shown in fig. This operation is carried out in the similar way as plain turning.The work is turned like stairs step, when it is made to rotate about the lathe axis, and the tool is fed parallel to the lathe axis. The straight turning produces a cylindrical surface by removing excess metal from the work piece. After facing the ends and drilling the centre, the job is carefully mounted between the centers using a lathe dog attached to the work piece. A properly ground right hand turning tool is used for this purpose. Tool is clamped on the tool post with the minimum overhang and is set with its cutting edge approximately at the lathe axis or slightly above it. For light cuts the tool may be inclined towards the headstock, but for heavy cuts the tool must be inclined towards the tailstock. The automatic feed is engaged to move the carriage to the desired length, then the feed is disengaged and the carriage is brought back to the starting position. The process is repeated until the job is finally finished.



STEP TURNING

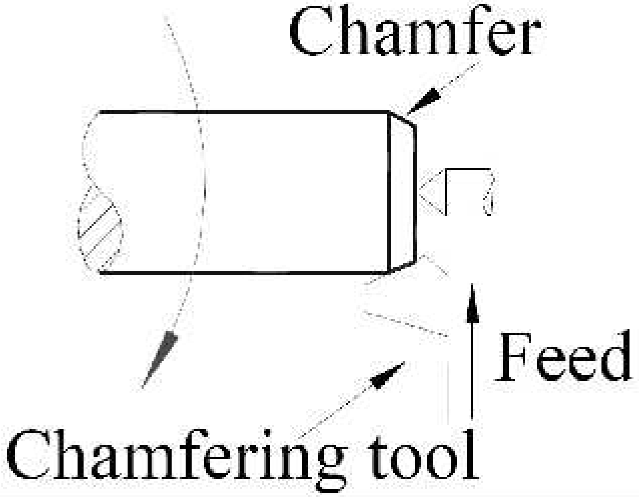
**Knurling:**

Knurling is the process of embossing a diamond shape pattern on the surface of a work piece. The purpose of knurling is to provide an effective gripping surface on a work piece, to prevent it from slipping when operated by hand. In some press fit work knurling is done to increase the diameter of a shaft. The operation is performed by a special knurling tool which consists of a set of hardened steel rollers in a holder with the teeth cut on their surface in a definite pattern. Knurls are available in coarse, medium and fine pitches.



**Chamfering:**

Chamfering is the operation of beveling the extreme end of a workpiece. This is done to remove the burrs, to protect the end of the work piece from being damaged and to have a better look. The operation may be performed after knurling, rough turning, boring, drilling. Chamfering is an



essential operation before thread cutting so that the nut may pass freely on the threaded workpiece.

**PROCEDURE:**

1. Prepare the tooling layout for the given workpiece.
2. Set the tool in their respective position of the tool post.
3. The workpiece is chucked and checked for the rotation.
4. Switch on the motor after selecting the proper speed
5. First by moving the cross slide facing operation is completed
6. After reducing the height of the work piece, turning operation is done.
7. Make the step turning operation as per the given dimension.
8. Chamfer the corners and check the dimensions.

**RESULT:**

The work of desired shape, size, and design is produced on lathe machine.

**PRECAUTIONS:**

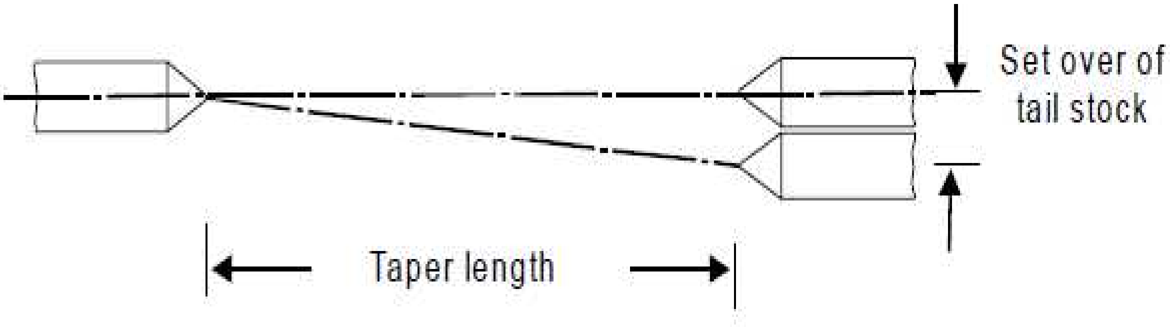
1. Correct dress is important, remove rings and watches, and roll sleeves above elbows.
2. Always stop the lathe before making adjustments.
3. Does not change spindle speeds until the lathe comes to a complete stop.
4. Handle sharp cutters, centers, and drills with care.
5. Remove chuck keys and wrenches before operating
6. Always wear protective eye protection.
7. Handle heavy chucks with care and protect the lathe ways with a block of wood when installing a chuck.
8. Know where the emergency stop is before operating the lathe.
9. Use pliers or a brush to remove chips and swarf, never your hands.

## EXPERIMENT NO-3

**AIM:-**Taper turning by tailstock offset method as per drawing.

**TOOLS AND EQUIPMENT:-**
1. Lathe machine
2. Single point cutting tool
3. Required job

**Theory:-**
This method is basically employed for turning small tapers on longer jobs and is confined to external tapers only. In this method, the tailstock is set over is calculated using Fig. by loosening the nut from its centre line equal to the value obtained by formula given below.



Tail stock set over = Taper length × Sine of half of taper angle

$(D – d) / 2 = l \times \sin (a/2)$

Where, D = is the diameter of the large end of cylindrical job,
*d* = is the diameter of the small end of cylindrical job, and
*l* = is the length of the taper of cylindrical job, all expressed in inches,
*a* = taper angle

When a part length of the job is to be given taper then tail stock set
= ((D – *d*)/2)) × (total length of the cylindrical job/length of taper)
= l × sin (*a*/2) × (total length of the cylindrical job/length of taper)

**Methods of offsetting the Tailstock:**

The tailstock may be offset by three methods:
1. By using the graduation on the end of tailstock.
2. By means of the graduated collar and feeler gage.
3. By means of a dial indicator.

**PROCEDURE:-**

1. Adjust the tailstock spindle to the distance it will be used in the machining setup and lock the tailstock spindle clamp.
2. Mount a dial indicator in the toolpost with the plunger in a horizontal position and on center.
3. Loosen the tailstock clamp nut.
4. Offset the tailstock the required amount.
5. With the tailstock adjusting setscrews, move the tailstock until the required offset is shown on the dial indicator.
6. Set up the cutting tool as for parallel turning.
7. Tighten the tailstock setscrew that was loosened, being sure that the indicator does not change.
7. Tighten the tailstock clamp null.
8. Check the taper for accuracy using a taper ring gage, if required.
9. Finish-turn the taper to the size and fit required.

**RESULT:-**

Our job will be ready by the method taper turning by tailstock offset method as per drawing.

**PRECAUTIONS:-**

1. Correct dress is important, remove rings and watches, and roll sleeves above elbows.
2. Always stop the lathe before making adjustments.
3. Does not change spindle speeds until the lathe comes to a complete stop.
4. Remove chuck keys and wrenches before operating
5. Always wear protective eye protection.
6. Know where the emergency stop is before operating the lathe.
7. Use pliers or a brush to remove chips and swarf, never your hands.

## EXPERIMENT-4

**AIM**: - TO CUT METRIC THREAD AS PER DRAWING

**TOOLS ANDEQUIPMENT:-**

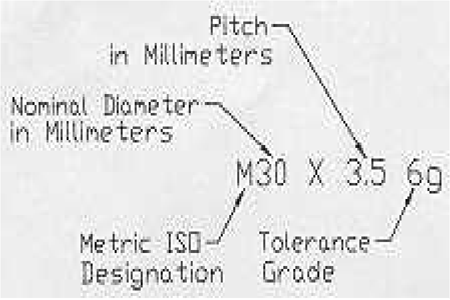
1. CUTTING TOOL
2. LATHE MACHINE
3. VERNIER CALIPER

**THEORY:**

Metric threading using a lathe with an inch lead screw isn't difficult, but instructions that cover the finer points are few and far between. It's a metric world and I now cut more metric threads than anything else. Chances are, if you're not cutting metric threads now, you will be sooner or later. My guess is sooner. With just a few extra gears you can cut any standard metric pitch on your small Logan, South bend or other quality lathe. The following explanation may seem a bit tedious and repetitive, but a solid understanding of the basics will help you solve any problems that might arise, for any lathe, gear train or pitch. Many parts in the machine shop will require the machinist to cut threads. As you have learned earlier in this module the most common form of thread is the V-form thread. Unified National or Standard Unified threads are the norm in the United States, but with the ever-growing influx of foreign manufactured products, especially in machine tools and automobiles, metric threads have become very popular. This unit will show you how to cut a metric thread.

Metric threads come in different forms. The System International (SI) metric thread form is very similar to the American National Standard thread form. The International Standard (ISO) was adopted to standardize metric thread forms.

Metric threads can be identified by the letter **M** preceding the major thread diameter, and the pitch (Figure 15-1).



The ISO metric thread has a 60-degree included angle. This is the same as the unified thread form. This makes cutting the metric thread form very similar to cutting the unified thread form. Metric thread may be cut on a standard lathe with the proper gear selection.

## PROCEDURE:-

1. Setup part between centers using a dead center in the tailstock. Make sure to use the proper lubricant on the dead center.
2. Check lathe setup for run out and taper.
3. Setup lathe to cut the first thread. Be ready to answer questions concerning half nut lever Engagement.
4. Instructor Have instructor check setup *
5. Take a light scratch cut. Insure that the pitch is correct.
6. Check pitch diameter using pitch micrometers or other thread measuring instrument.
7. Continue prior procedures for the rest of the threads. Use a split nut when clamping on the finished thread end.
8. DE burr thread crests and starts. If filing thread crest be careful not to take too much off.
9. Inspect part and turn in for grading.

## RESULT:

The work of desired shape, size, and design is produced on lathe machine.

## PRECAUTIONS:

1. correct dress is important, remove rings and watches, and roll sleeves above elbows.
2. Always stop the lathe before making adjustments.
3. Does not change spindle speeds until the lathe comes to a complete stop?
4. Handle sharp cutters, centers, and drills with care.
5. Remove chuck keys and wrenches before operating
6. Always wear protective eye protection.
7. Handle heavy chucks with care and protect the lathe ways with a block of wood when installing a chuck.
8. Know where the emergency stop is before operating the lathe.
9. Use pliers or a brush to remove chips and swarf, never your hands.

## EXPERIMENT NO-5

**AIM: -** TO PREPARE MOULD OF GIVEN PATTERN REQUIRING CORE AND TO CAST IT IN ALUMINIUM

**APPRATUS REQURIED:-**

1. Molding board
2. Shovel
3. Rammer
4. Pattern
5. Riser pin
6. Aluminum
7. Furnanance
8. Molding flask
9. Core boxes
10. Silica sand

THEORY:-

**Aluminum – Castings**

**Introduction**

Aluminum casting processes are classified as Ingot casting or Mould casting. During the first process, primary or secondary aluminums is cast into rolling ingot (slab), extrusion ingot (billet) and wire bar ingot which are subsequently transformed in semi- and finished products.

The second process is used in the foundries for producing cast products. This is the oldest and simplest (in theory but not in practice) means of manufacturing shaped components.

This section describes exclusively Mould casting which can be divided into two main groups:

1.Sandcasting

2.Diecasting

Other techniques such as "lost foam" or "wax pattern" processes are also used but their economical importance is considerably lower than both listed techniques.

## Sand casting

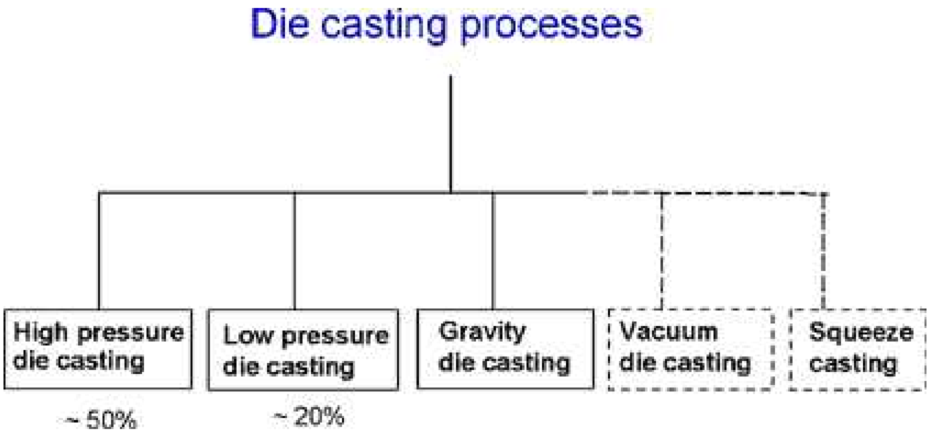
In sand casting, re-usable, permanent patterns are used to make the sand moulds. The preparation and the bonding of this sand mould are the critical step and very often are the rate-controlling step of this process. Two main routes are used for bonding the sand moulds:

The "green sand" consists of mixtures of sand, clay and moisture.
The "dry sand" consists of sand and synthetic binders cured thermally or chemically.
The sand cores used for forming the inside shape of hollow parts of the casting are made using dry sand components.
This versatile technique is generally used for high-volume production

**Die-casting:-**

In this technique, the mould is generally not destroyed at each cast but is permanent, being made of a metal such as cast iron or steel. There are a number of die casting processes, as summarized in Figure 4. High pressure die casting is the most widely used, representing about 50% of all light alloys casting production. Low pressure die casting currently accounts for about 20% of production and its use is increasing. Gravity die casting accounts for the rest, with the exception of a small but growing contribution from the recently introduced vacuum die casting and squeeze casting process.

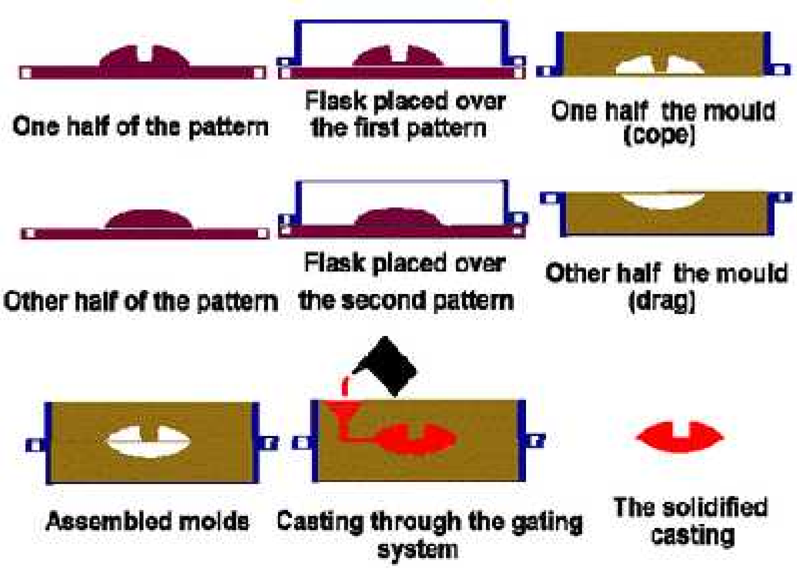
*Figure 3: Classification of die casting processes*



PROCEDURE:-

1. Take a wooden pattern
2. Patterns mounted on core pins

3. Core boxes core halves
4. Cope half of mould is assembled by securing the cope pattern plate in the plask with aligning pins and attaching wrests forms the sprue and sires
5. Ram san the plane and insert are removed
6. The drag half is prepared in similar way
7. Pattern flask and bottom board inverted and pattern is removed
8. The core is set in place within the drag


One half of the pattern
Flask placed over the first pattern
One half the mould (cope)
Other half of the pattern
Flask placed over the second pattern
Other half the mould (drag)
Assembled molds
Casting through the gating system
The solidified casting

PRECAUTION:-

1. Place pattern in half of the core prints with pin.
2. Protect flask from subjected pressure
3. Handle mould very carefully.
4. The joint should be accurate and well finished.

CONCLUSION:-

Thus, prepared a mould of a given pattern successfully

# UNIVERSITY OF ENGINEERING & MANAGEMENT, JAIPUR
## Lab Manual

**Title of Course: Applied Mechanics Lab**
**Course Code: ME-393**
**L-T-P scheme: 0-0-3** **Course Credit: 2**

**Objectives:**

**1.** To learn and understand system of bodies to external forces, stress & strain in body.
**2.** Basic concept of statics, kinematics dynamics and material mechanics.
**3.** To provide an understanding of the design aspects of different materials.
**4.** To provide knowledge of the friction, torsion, pure bending, beam deflection.

**Learning Outcomes:** The students will have a detailed knowledge of the concepts of applied mechanics. Analyze kinematics, kinematics of both particles and rigid bodies systems and apply them to practical engineering system design and development. Analyze force system and apply them to practical engineering system design and development. Student can carry out stress and strain analysis of beams and simple structures apply them to practical engineering system. Student can carry out torsion, pure bending and shearing stress and apply them to practical engineering system. Upon the completion of Applied Mechanics Lab course, the student will be able to:

- **Understand** and implement basic concepts of applied mechanics.
- Analyze kinematics, kinematics of both particles and rigid bodies systems and apply them to practical engineering system design and development**.**
- **Understand** the force system and apply them to practical engineering system design and development
- **Analyze** and carry out stress and strain analysis of beams and simple structures apply them to practical engineering system
- Understand torsion, pure bending and shearing stress and apply them to practical engineering system.
- Deploy applied mechanics knowledge to solve the practical engineering problem of products and system design and development

**Course Contents:**

Experiments **that must be done in this course are listed below:**
Experiment No.1: Determining spring stiffness under tension and compressive loads.
Experiment No.2: To Study various types of Strain Gauges.
Experiment No 3: Torsion Test.
Experiment No.4: Brinnel Hardness Test.
Experiment No.5: Rockwell Hardness Test.
Experiment No.6: Experiments on friction. Determination of coefficient of friction
Experiment No.7: To Study The Universal Testing Machine (U.T.M.)
Experiment No.8: To determine tensile test on a metal.
Experiment No.9: Compression test of ductile and brittle materials on UTM

**Text Book:**

1. J.L. Meriam, L.G. Kraige, Engineering Mechanics, Wiley
2. Hannah J & Hillier M.J, Applied Mechanics, Longman

**Recommended Equipments/Systems/Software Requirements:**

**1.** U.T.M., Torsion testing machine, hardness testing machine.
**2.** Strain gauge, spring stiffness test apparatus, specimens.

**Experiment No: 1 determining spring stiffness under tension and compressive loads**
**Aim: To determine the stiffness of the spring and modulus of rigidity of the spring wire**

**APPARATUS: - (**i) Spring testing machine. (ii) A spring (iii) Vernier caliper, Scale.
(iv) Micrometer.

**DIAGRAM:-**

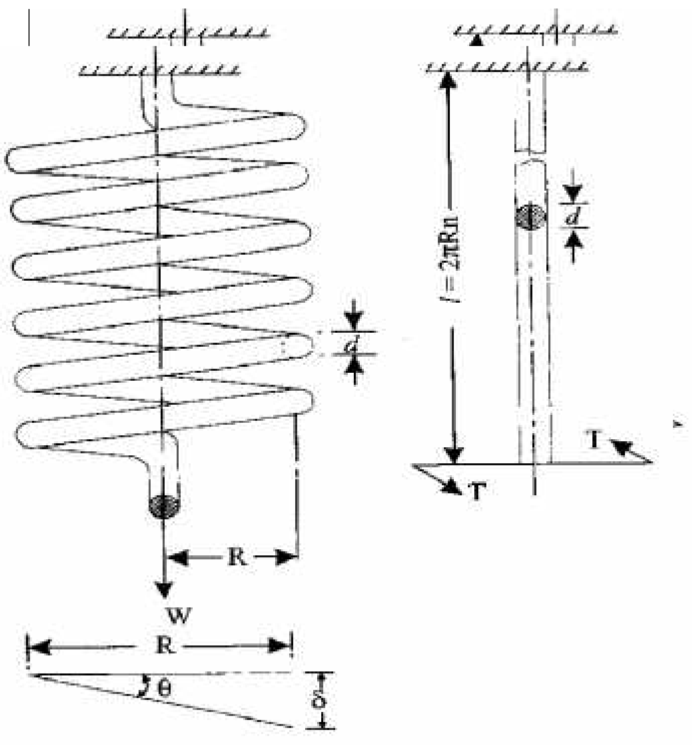


**Fig. 14.1.** Close-coiled helical spring.

**THEORY: -** Springs are elastic member which distort under load and regain their original shape when load is removed. They are used in railway carriages, motor cars, scooters, motorcycles, rickshaws, governors etc. According to their uses the springs perform the following Functions:

1) To absorb shock or impact loading as in carriage springs.
2) To store energy as in clock springs.
3) To apply forces to and to control motions as in brakes and clutches.
4) To measure forces as in spring balances.
5) To change the variations characteristic of a member as in flexible mounting of motors.

The spring is usually made of either high carbon steel (0.7 to 1.0%) or medium carbon alloy steels. Phosphor bronze, brass, 18/8 stainless steel and Monel and other metal alloys are

Several types of spring are available for different application. Springs may classify as helical springs, leaf springs and flat spring depending upon their shape. They are fabricated of high shear strength materials such as high carbon alloy steels spring form elements of not only mechanical system but also structural system. In several cases it is essential to idealise complex structural systems by suitable spring.

**PROCEDURE:**

**1)** Measure the diameter of the wire of the spring by using the micrometer.
2) Measure the diameter of spring coils by using the vernier caliper
3) Count the number of turns.
4) Insert the spring in the spring testing machine and load the spring by a suitable weight and note the corresponding axial deflection in tension or compression.
5) Increase the load and take the corresponding axial deflection readings.
6) Plot a curve between load and deflection. The shape of the curve gives the stiffness of the spring.

**OBESERVATION:**

Least count of micrometer = ……mm
Diameter of the spring wire, d =………mm (Mean of three readings)
Least count of vernier caliper = ……mm
Diameter of the spring coil, D = ……mm (Mean of three readings)
Mean coil diameter, Dm = D - d……mm
Number of turns, n =

**OBESERVATION TABLE:**

| **S.NO** | **Load**<br>**W**<br>**(N)** | **Deflection**<br>**(δ)**<br>**(mm)** | **Stiffness K = W / δ**<br>**N / mm** |
|---|---|---|---|
| 1 | | | |
| 2 | | | |
| 3 | | | |
| 4 | | | |
| 5 | | | |

Mean k = ……

**Modulus of rigidity $C=8W D^3_m n/ \delta d^4$**

**Spring Index = $D_m/D$**

**RESULT:** The **value** of spring constant k of closely coiled helical spring is found to be------------ **N / mm**

**PRECAUTION: - 1)** The dimension of spring was measured accurately.
2) Deflection obtained in spring was measured accurately

-------------------------------------------------------------------------------------------------------------
---

**Experiment No: 2** To Study various types of strain Gauges.
**Aim**: To Study various types of strain Gauges.

**THEORY : -** A strain Gauge may be defined as any instrument or device that is employed to measure the linear deformation over a given gauge length, occurring in the material of a structure during the loading of structures. This definition is quite broad. In fact it covers the range of instruments included between the linear scale & the precise optical & electrical gauges now available. The many types of strain gauges available are quite varied both in applications & in the principle invalid in their magnification, systems. Depending upon the magnification system the strain gauges may be classified as follows:

**1) Mechanical**
A. Wedge & screw
B. Lever – simple & compound
C. Rock & pinion
D. Combination of lever & rack & pinion
E. Dial indicators

**2) Electrical**
A. Inductance
B. Capacitance
C. Piezoelectric

**Accuracy & repeatability** -: Sensitive does not ensure accuracy. Usually the very sensitive instruments are quite prone to error unless they are employed with utmost care. Before selecting a particular type of gauge following factors must also be carefully evaluated.

1) Readability
2) Ease of mounting
3) Required operator skill
4) Weight
5) Frequency response
6) cost

1) Mechanical Strain Gauges:-

a) Wedge & screw :-

The wedge gauge is simply a triangular plate with its longer sides related at 1:10 slope when inserted between two shoulders dipped to the test specimen, extension could be detected nearest 0.05 mm .A single screw extensometer which is one of the pioneer instruments used for measurement of strain. The magnification in this instrument is accomplished solely by a screw micrometer a measures the relative motion of two coaxial tubes

1. Magnetic

2. Acoustical

3. Pneumatic

4. Scratch type

5. Photo stress gauge

Characteristic of a strain gauge:-

A strain gauge has the following four basic characteristics

1) Gauge length: - The gauge size for a mechanical strain gauge is characterized by the distance between two knife edges in contact with the specimen & by width of a movable knife edges non linear strum which should be as small as possible in that case.
2) Sensitivity: - It is the smallest value of strain which can be read on the scale associated with strain gauge .Sensitivity can be defined in two ways:-

$$\text{i) Deformation sensitivity} = \frac{\text{Smallest reading of scale}}{\text{Multiplication factor}}$$

$$\text{ii) Strain sensitivity} = \frac{\text{Deformation sensitivity}}{\text{Base length}}$$

3) Range: - This represents the maximum strain which can be recorded without resetting or replacing the strain gauge. The range & sensitivity are

1) Simple Mechanical lever magnification:-
The simple lever strain gauge gains its magnification factors by a suitable positioning of fulcrum cap's multiplying divider is an important extensiomeus of this category. The magnification of this type of gauge is unlimited. The gauge length of cap's divider is 5cm & strain is magnified 10: 1 on graduated scale.

2) Compound Magnification System:-
Two commercially available gauges which utilize the compound magnification are illustrated by Barry gauge & tinusis oisen strain gauge. The Barry strain gauge consists of frame a with two conically painted contact points. One point b is rigidly fixed to frame while other c is provided from a frame & is internal with a lever armed which alone magnifies the strain about 5.5. A screw micrometer or dial indicator is used to measure the motion of arm, thus permitting measurements of strain to nearest 0.005 m with a 0.025mm micrometer.

3) Compound lever Magnification:-
Two gauges of this category are Huggenberger strain gauge & parter lipp strain gauge. In these instruments the magnification system is composed of two or more simple levers in series. They have relatively small size & high magnification factor.

4) Mechanical by rack & pinion:-

The rack & pinion principle alone with various types of gear train is employed in gauge in which the magnification system is incorporated in an indicating dial. In general a dial indicator consists of an encased grain train actuated by a rack cut in spindle which follows the motion to be measured. A spring imposes sufficient spindle force to maintain a reasonably uniform & positive contact with the moving part. The gear train terminates with a light weight pointer which indicator spindle travel on a graduated dial. Lost motion in gear train is minimized by +ve force of a small coil spring the dial gauge extensometer is the most popular gauge of this type used in a material testing laboratory. Dial gauge indicator are frequently attached permanently to a structure to indicate the deflection one deflection on deformation obtained under working condition.

3**) Acoustical strain gauge**:-

The vibrating wire or acoustical gauge consists essentially of a steel wire tensioned between two supports a predetermined distance apart. Vibration of the distance alters the natural frequency of vibration of the wire & thus change in frequency may be correlated with the change in strain causing an electro – magnet adjacent to the wire may be used to set the wire in vibration & this wire movement will then generate on oscillating electrical signal. The signal may be compared with the pitch adjustable standard wire, the degree of adjustment necessary to match of two signal frequencies being provided by a tensioning Screw on the slandered wove calibration of this screw allows direct determination of change of length of a measuring gauge to be made once the standard gauge has been tuned to match the frequency of measuring wire.

The visual display produced is a cko renders adjustment easier. Tuning is now more usually accomplished by feeding the two signals in to two pours of plates of an oscillogram & making use of the luscious figure formation to balance the frequencies. Matching of tones is simplified & made more accurate by tuning out the beats with results when the vibration frequencies of two were are nearly the lame.

The fundamental frequency of stretch wire

$$f = \frac{1}{2\,L} \quad \frac{p}{m} \quad \frac{p}{2l} \quad \frac{e\,l\,A}{L}$$

A = Cross sectional area

E = Young’s modulus of were

h = length of vibrating were

m = mass per unit length of were

p = tensioning force is were

w = increment in length of vibrating were

-----------------------------------------------------------------------------------------------------------

**Experiment No: 3 To conduct torsion test on mild steel or cast iron specimens to find out modulus of rigidity**

**AIM: - Torsion test on mild steel rod.**

**APPARATUS: -1.Torsion testing**

**2. Twist meter for measuring angles of twist**

**3. A steel rule and Vernier Caliper or micrometer.**

**DIAGRAM:-**

**THEORY: -**

A torsion test is quite instrumental in determining the value of modulus of rigidity of a metallic specimen. The value of modulus of rigidity can be found out thought observations made during the experiment by using the torsion equation

$$\frac{T}{I_p} = \frac{C\theta}{L} = \frac{q}{r}$$

Where, T = Torque applied,

$I_p$ = Polar moment of inertia

C = Modulus of rigidity,

$\theta$ = Angle of twist (radians), and

L = Length of the shaft

q = Shear stress

r = Distance of element from center of shaft

**PROCEDURE:-**

1. Select the driving dogs to suit the size of the specimen and clamp it in the machine by adjusting the length of the specimen by means of a sliding spindle.
2. Measure the diameter at about three places and take the average value.
3. Choose the appropriate range by capacity change lever
4. Set the maximum load pointer to zero.
5. Set the protector to zero for convenience and clamp it by means of knurled screw.
6. Carry out straining by rotating the hand wheel in either direction.
7. Load the machine in suitable increments.
8. Then load out to failure as to cause equal increments of strain reading.
9. Plot a torque- twist (T- θ) graph.
10. Read off co-ordinates of a convenient point from the straight line portion of the torque twist (T- θ) graph and calculate the value of C by using relation.

**OBESERVATION:-**

$$C= TL/\Theta I_P$$

Gauge length of the specimen, L = ………

Diameter of the specimen, d = ………

Polar moment of inertia, $I_P = \pi d^4 / 32$ = …………

**OBESERVATION TABLE:-**

| **Torque (T)** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Angle of twist(θ)in 'radians'** | | | | | | | | | | | | | | | |
| **Modulus of rigidity (C) N/mm$^2$** | | | | | | | | | | | | | | | |

**RESULT :-** I) **Modulus of rigidity of mild steel rod is ------------- N/mm2**

**II) Modulus of rigidity of Aluminum rod is ------------- N/mm2**

**PRECAUTION:-** 1) Measure the dimensions of the specimen carefully

2) Measure the Angle of twist accurately for the corresponding value of Torque.

---

Experiment No. 4 **Brinell hardness Test**

Aim: - To determine the hardness of the given specimen using Brinell hardness test.

**Specimen and apparatus**

Brinell hardness tester
Specimen
Ball indenter.

**Theory**

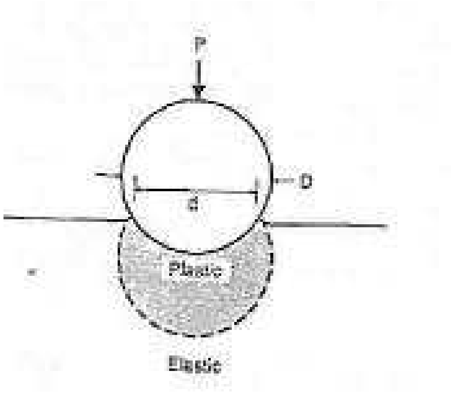
Hardness of a material is generally defined as Resistance to the permanent indentation under static and dynamic load. When a material is required to use under direct static or dynamic loads, only indentation hardness test will be useful to find out resistance to indentation.

In Brinell hardness test, a steel ball of diameter (D) is forced under a load (F) on to a surface of test specimen. Mean diameter (d) of indentation is measured after the removal of the load (F).

**Brinell Test**

In a standard Brinell test 10 mm diameter hardened steel ball is forced to penetrate the material by 3000 kgf for steels and cast irons. The load and ball diameter selection is important depending on the hardness of materials and 500 kgf is used for softer materials with the same ball diameter. Keeping the ratio of load *P* to the square of diameter *D2* constant (30 for steels and cast irons and 5 for soft metals and alloys), different load and ball diameter combinations can be selected and used in Brinell hardness testing. The Brinell Hardness Number (BHN) is obtained by dividing the applied force P, in kgf; by the curved surface area of the indentation, which is actually a segment of sphere. The geometry of indentation is given in Figure-1, and the hardness is determined according to the relationship,

$$B.H.N. = \frac{2f}{\pi \ \{D - \sqrt{D^2 - d}\}^2}$$



**Observation**

1. Take average of five values of indentation of each specimen. Obtain the hardness number from equation.
2. Compare Brinell and Rockwell hardness tests obtained.

**Procedure**

1. Load to be applied for hardness test should be selected according to the expected hardness of the material. However test load shall be kept equal to 30 times the square of the diameter of the ball (diameter in mm)

$$F=30.D^{2}$$

Where ball diameter, generally taken as 10 mm.

For guidelines hardness range for standard loads given below

| Ball diameter | Load (kg) | Range of Brinell hardness |
|---|---|---|
| 10 | 3000 | 96 to 600 |
|  | 1500 | 48 to 300 |

2.Apply the load for a minimum of 15 seconds to 30 seconds. [If ferrous metals are to be tested time applied will be 15 seconds and for softer metal 30 seconds]

3. Remove the load and measure the diameter of indentation nearest to 0.02 mm using microscope (projected image)

4. Calculate Brinell hardness number (HB). As per IS: 1500.
5. Brinell hardness number

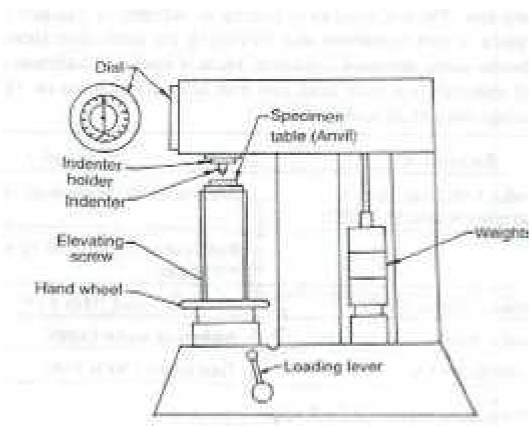
$$B.H.N. = \frac{2f}{\pi \ \ \{D - \sqrt{D^{2} - d}\}^{2}}$$

Where D is the diameter of ball indenter and d is the diameter of indentation.

Hardness numbers normally obtained for different materials are given below (under 3000 kg and 10 mm diameter ball used)

|  |  |
|---|---|
| Ordinary steels medium carbon | 100 to 500 |
| Structural steel | 130 to 160 |
| Very hard steel | 800 to 900 |

Note: Brinell test is not recommended for the materials having HB over 630.
It is necessary to mention ball size and load with the hardness test when standard size of ball and load are not used. Because indentation done by different size of ball and load on different materials are not geometrically similar. Ball also undergoes deformation when load is applied. Material response to the load is not same all the time.
6. Brinell hardness numbers can be obtained from tables 1 to 5 given in IS: 1500, knowing diameter of indentation, diameter of the ball and load applied.



**Precautions**

1. Thickness of the specimen should not be less than 8 times the depth of indentation to avoid the deformation to be extended to the opposite surface of a specimen.
2. Indentation should not be made nearer to the edge of a specimen to avoid unnecessary concentration of stresses. In such case distance from the edge to the centre of indentation should be greater than 2.5 times diameter of indentation.
3. Rapid rate of applying load should be avoided. Load applied on the ball may rise a little because of its sudden action. Also rapidly applied load will restrict plastic flow of a material, which produces effect on size of indentation.
4. Surface of the specimen is well polished, free from oxide scale and any foreign material.

**Result**:

The Brinell hardness number of the specimen is - - ----

…………………………………………………………………………………………….

**Experiment No. 5:- Rockwell Hardness test**

**Aim: -** To determine the Hardness of the given Specimen using Rockwell hardness test.

**Materials and equipments required**

- Rockwell hardness testing machine.
- Black diamond cone indenter
- Hard steel specimen.

**Theory**

Rockwell test is developed by the Wilson instrument co U.S.A in 1920.

This test is an indentation test used for smaller specimens and harder materials. The test is subject of IS: 1586.In this test indenter is forced into the surface of a test piece in two operations, measuring the permanent increase in depth of an indentation from the depth increased from the depth reached under a datum load due to an additional load.

Measurement of indentation is made after removing the additional load. Indenter used is the cone having an angle of 120 degrees made of black diamond.

**Rockwell Test**

In the **Rockwell** test, a diamond cone or a hard steel ball is employed as the indenter depending on the hardness of materials. Diamond cone or *Brale* indenter with cone angle of $120^0$ is used to test hard materials and the balls of sizes between 1.6 mm (1/16") and 12.7 mm (1/2") are used in testing softer materials.

Rockwell tests differ from other indentation hardness tests in that the depth of indentation determines the hardness rather than the indentation size (see Figure-3). Therefore, surface condition of specimens is very important in Rockwell testing because of its high dependency on the accuracy in indentation depth measurements. In order to establish a reference position a *minor load* of 10 kgf. Is first applied, and the major load is then applied. Additional penetration due to *major load* is measured and readings are obtained from a calibrated scale (dial) directly, which has a maximum value of 100, depending on the depth of penetration. Most commonly used Rockwell hardness scales are given in Table-l with typical applications. The hardness numbers are designated *HRX*, where *X* indicates the scale used (i.e. 50 HRC for 50 points on the C scale of dial). It should be noted that a Rockwell hardness number is meaningless unless the scale is not specified.

**HR = E-e**

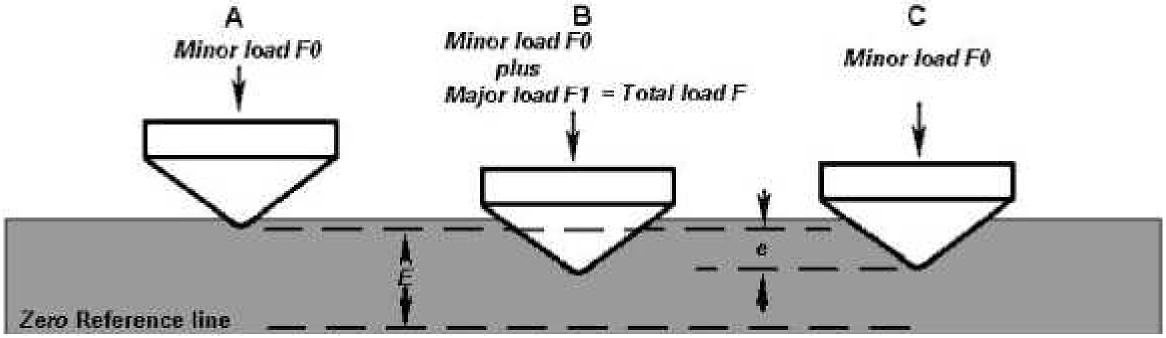


Figure: Increasing depth of penetration in the Rockwell test
Table-I: Commonly used Rockwell hardness scales

| Scale X | Indenter Type | I Major Load | Typical Applications |
|---|---|---|---|
| A | Diamond Brale | 60 | Tool Materials |
| D | Diamond Brale | 100 | Cast Irons, Sheet Steels |
| C | Diamond Brale | 150 | Hardened steels and cast irons, Ti alloys |
| B | 1/16" Diameter Ball | 100 | Annealed steels, Cu and Al alloys |
| E | 1/8" Diameter Ball | 100 | Al and Mg alloys, reinforced polymers |
| F | 1/16" Diameter Ball | 60 | Soft sintered products |
| M | 1/4" Diameter Ball | 100 | Very soft metals, polymers |
| R | 1/2" Diameter Ball | 60 | Very soft metals, polymers |

Since the deformations caused by an indenter are of similar magnitude to those occurring at the ultimate tensile strength in a tension test, some empirical relationships have been established between hardness and engineering ultimate tensile strength of metals and alloys. For example, for steels UTS can be **roughly estimated** from Brinell hardness as follows:
UTS (in MPa) = 3.45 x BHN

**Procedure**

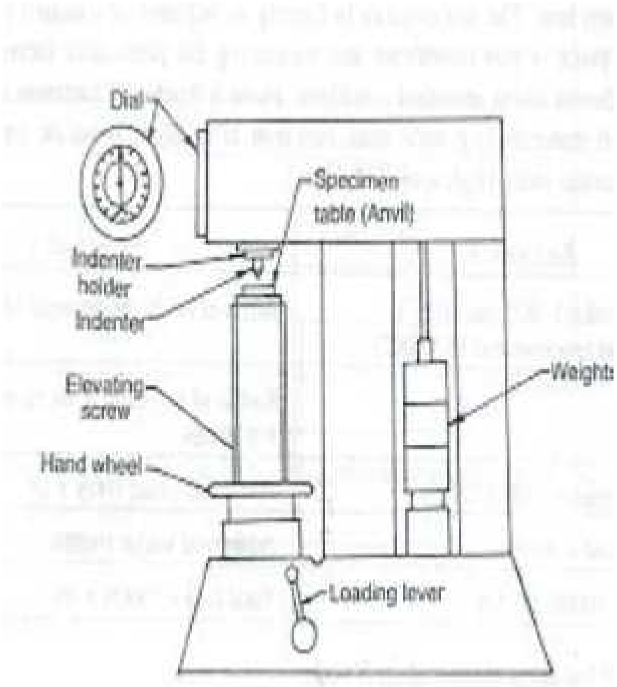
1. Examine hardness testing machine (fig.1).

2. Place the specimen on platform of a machine. Using the elevating screw raise the platform and bring the specimen just in contact with the ball. Apply an initial load until the small pointer shows red mark.
3. Release the operating valve to apply additional load. Immediately after the additional load applied, bring back operating valve to its position.
4. Read the position of the pointer on the C scale, which gives the hardness number.
5. Repeat the procedure five times on the specimen selecting different points for indentation.

**Observation**

1. Take average of five values of indentation of each specimen. Obtain the hardness number from the dial of a machine.
2. Compare Brinell and Rockwell hardness tests obtained.

**Figure .1**



**Precautions**

1. Thickness of the specimen should not be less than 8 times the depth of indentation to avoid the deformation to be extended to the opposite surface of a specimen.
2. Indentation should not be made nearer to the edge of a specimen to avoid unnecessary concentration of stresses. In such case distance from the edge to the centre of indentation should be greater than 2.5 times diameter of indentation.
3. Rapid rate of applying load should be avoided. Load applied on the ball may rise a little because of its sudden action. Also rapidly applied load will restrict plastic flow of a material, which produces effect on size of indentation.
4. Surface of the specimen is well polished, free from oxide scale and any foreign material.

**Result**

Rockwell hardness of given specimen is **……………..**

**…………………………………………………………………………………………….**

**Experiment No.6: Experiments on friction. Determination of coefficient of friction**

**AIM:-**To find the Coefficient of Friction between two surfaces.

**APPARATUS: -** Inclined Plane with pulley, weights, Trolleys, weight pan etc.

**THEORY:**

Friction force is developed whenever there is a motion or tendency of motion of one body with respect to the other body involving rubbing of the surfaces of contact. Friction is therefore a resistance force to sliding between two bodies produced at the common surfaces of contact.

Friction occurs because no surface is perfectly smooth, however flat it may appear. On every surface there are ‘microscopic hills and valleys’ and due to this the surfaces get interlocked making it difficult for one surface to slide over the other. During static state the friction force developed at the contact surface depends on the magnitude of the disturbing force. When the body is on the verge of motion the contact surface offers maximum frictional force called as ‘Limiting Frictional Force’.

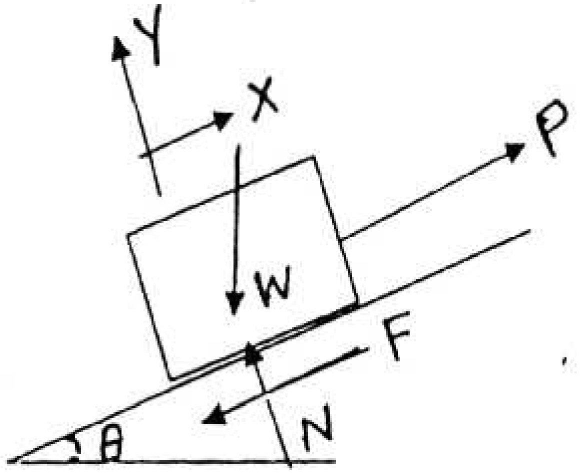
In 1781 the French Physicist Charles de Coulomb found that the limiting frictional force did not depend on the area of contact but depends on the materials involved and the pressure (normal reaction) between them.

Thus frictional force F α N

or F = μsN

Here μs is the coefficient of static friction, a term introduced by Coulomb. The value of lies between 0 and 1 and it depends on both the surfaces of contact.

Coefficient of static friction ‘μs’ between two surfaces can be found out experimentally by two methods, viz. Angle of Repose method and Friction Plane method.



Applying Condition of Equilibrium

$$\Sigma F_x = 0$$

$$P - W\sin\theta - \mu_S N = 0 \qquad \text{.........} \ (1)$$

$$\Sigma F_y = 0$$

$$N - W\cos\theta = 0 \qquad \text{.........} \ (2)$$

From equations (1) and (2) we get

$$P - W\sin\theta - \mu_S(W\cos\theta) = 0$$

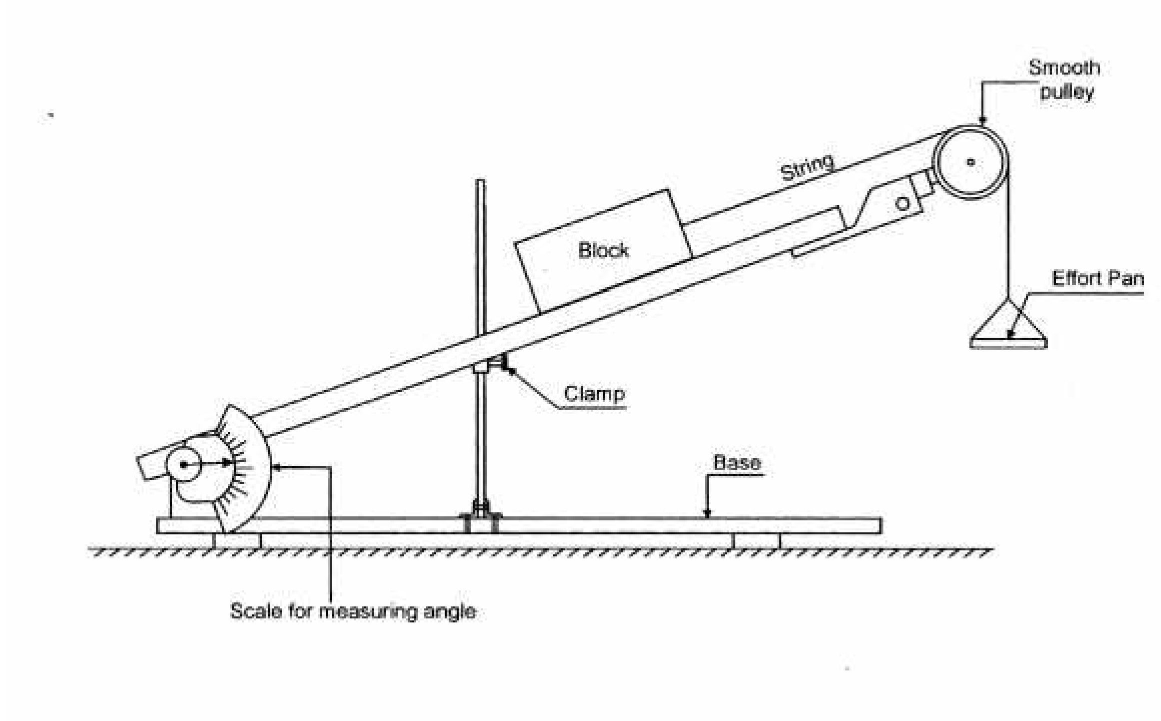
$$\text{or} \quad \boxed{\mu_S = \frac{P - W\sin\theta}{W\cos\theta}}$$

**OBSERVATION TABLE:**

| Type of Surface | Weight in trolley (W) | Angle of the Plane (Θ) | Weight in pan (P) | Coefficient of friction (μ) | Average (μ) | Angle of repose, φ | |
|---|---|---|---|---|---|---|---|
| | | | | | | Analytical | Experimental |
| Wood | | | | | | | |
| | | | | | | | |
| Metal | | | | | | | |
| | | | | | | | |

The minimum angle of an inclined plane at which a body kept on it slides down the plane without the application of any external force is known as Angle of Repose. It is denoted by letter φ.

Angle of repose, $\varphi = \tan^{-1} \mu_S$



**FRICTION PLANE APPARATUS**

**PROCEDURE**

1. Set the inclined plane with glass top at some angle with the horizontal. Note the inclination θ of the plane on the quadrant scale. Take a box of known weight, note its bottom surface (whether surface is soft wood, or sand paper, or card board etc,) and weight W (weight of box+ weights in the box).

2. Tie a string to the box and passing the string over a smooth pulley, attach an effort pan to it.

3. Slowly add weights in the effort pan. A stage would come when the effort pan just slides down pulling the box up the plane. Using fractional weights up to a least count of 5 gm, find the least possible weight in the pan that causes the box to just slide up the plane. Note the weight in the effort pan. This is force 'P'.

4. Repeat the above steps 1 to 3 by changing the weights in the box for two more sets of observations.

**RESULT:-**

1. The coefficient of friction between mica and wood is ------------

2. The coefficient of friction between mica and metal is ------------

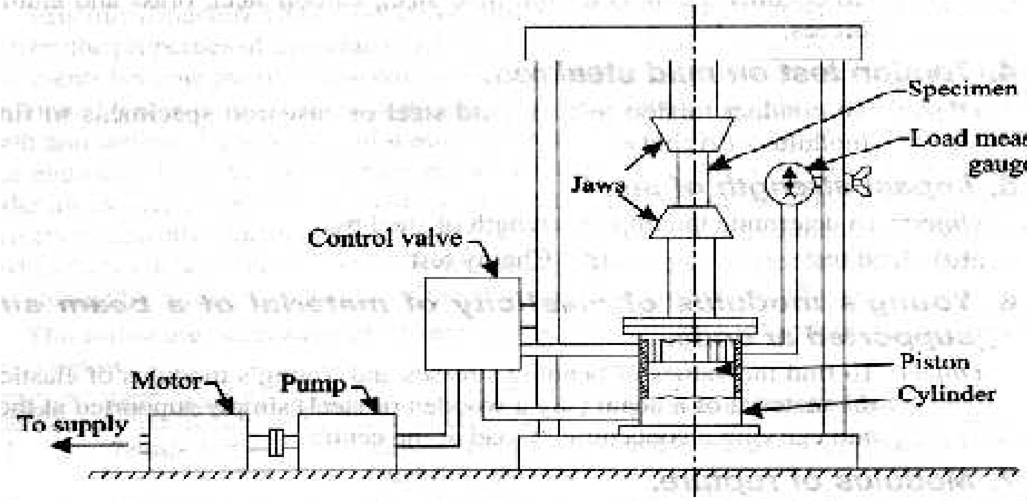
---------------------------------------------------------------------------------------------------

**Experiment No.7:** Study of Universal Testing Machine (U.T.M.)

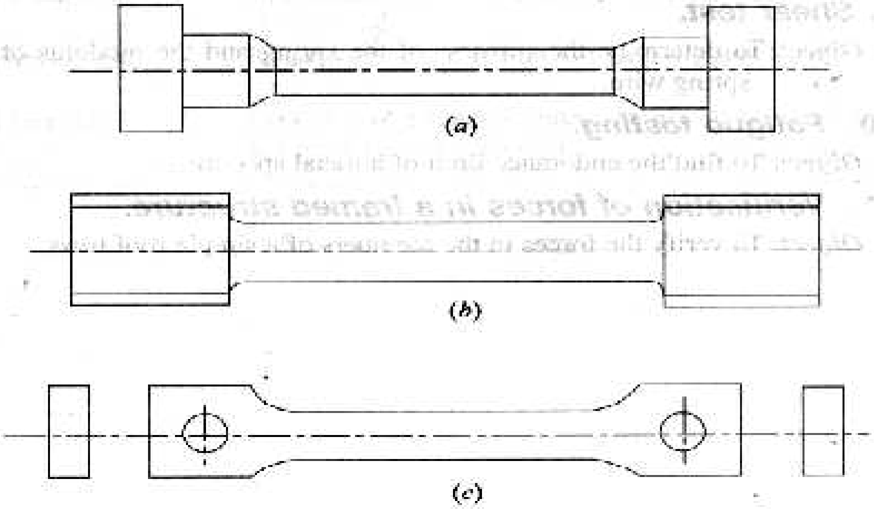
**AIM: -** To Study of Universal Testing Machine (U.T.M.)

**OBJECT**: - To Study the various component parts of the Universal Testing Machine (U.T.M.) & test procedures of various practical's to be performed.

**APPARATUS**: - Universal Testing Machine with all attachment i.e. shears test attachment, bending attachment, tension grips, compression test attachment etc.



• Fig. 1. Tenstile testing machine.


(a)

(b)

(c)

Fig. 2. Mild steel specimens.

**THEORY:** - The Universal Testing Machine consists of two units.
1) Loading unit
2) Control panel.

**LOADING UNIT:-**

It consists of main hydraulic cylinder with robust base inside. The piston which moves up and down. The chain driven by electric motor which is fitted on left hand side. The screw column maintained in the base can be rotated using above arrangement of chain. Each column passes through the main nut which is fitted in the lower cross head.

The lower table connected to main piston through a ball & the ball seat is joined to ensure axial loading. There is a connection between lower table and upper head assembly that moves up and down with main piston. The measurement of this assembly is carried out by number of bearings which slides over the columns.

The test specimen each fixed in the job is known as 'Jack Job'. To fix up the specimen tightly, the movement of jack job is achieved helically by handle.

It consists of oil tank having a hydraulic oil level sight glass for checking the oil level. The pump is displacement type piston pump having free plungers those ensure for continuation of high pressure. The pump is fixed to the tank from bottom. The suction & delivery valve are fitted to the pump near tank Electric motor driven the pump is mounted on four studs which is fitted on the right side of the tank. There is an arrangement for loosing or tightening of the valve. The four valves on control panel control the oil stroke in the hydraulic system. The loading system works as described below.

The return valve is close, oil delivered by the pump through the flow control valves to the cylinder & the piston goes up. Pressure starts developing & either the specimen breaks or the load having maximum value is controlled with the base dynameters consisting in a cylinder in which the piston reciprocates. The switches have upper and lower push at the control panel for the downward & upward movement of the movable head. The on & off switch provided on the control panel & the pilot lamp shows the transmission of main supply.

**METHOD OF TESTING:-**

**Initial Adjustment: -** before testing adjust the pendulum with respect to capacity of the test i.e. 8 Tones; 10 Tones; 20 Tones; 40 Tones etc.

For ex: - A specimen of 6 tones capacity gives more accurate result of 10 Tones capacity range instead of 20 Tones capacity range. These ranges of capacity are adjusted on the dial with the help of range selector knob. The control weights of the pendulum are adjusted correctly. The ink should be inserted in pen holder of recording paper around the drum & the testing process is started depending upon the types of test as mentioned below.

**TENSION TEST:-**

Select the proper job and complete upper and lower check adjustment. Apply some Greece to the tapered surface of specimen or groove. Then operate the upper cross head grip operation handle & grip the upper end of test specimen fully in to the groove. Keep the lower left valve in fully close position. Open the right valve & close it after lower table is slightly lifted. Adjust the lower points to zero with the help of adjusting knob. This is necessary to remove the dead weight of the lower table. Then lock the jobs in this position by operating job working handle. Then open the left control valve. The printer on dial gauge at which the specimen breaks slightly return back & corresponding load is known as breaking load & maximum load is known as the ultimate load.

**COMPRESSION TEST:-**

Fix upper and lower pressure plates to the upper stationary head & lower table respectively. Place the specimen on the lower plate in order to grip. Then adjust zero by lifting the lower table. Then perform the test in the same manner as described in tension test.

**FLEXURAL OR BENDING TEST:-**

Keep the bending table on the lower table in such a way that the central position of the bending table is fixed in the central location value of the lower table. The bending supports are adjusted to required distance.

Stuffers at the back of the bending table at different positions. Then place the specimen on bending table & apply the load by bending attachment at the upper stationary head. Then perform the test in the same manner as described in tension test.

**BRINELL HARDNESS TEST:-**

Place the specimen on the lower table & lift it up slightly. Adjust the zero fixed value at the bottom side of the lower cross head. Increase the load slowly ultimate load value is obtained. Then release the load slowly with left control valve. Get the impression of a suitable value of five to ten millimeter on the specimen & measure the diameter of the impression correctly by microscope & calculate Brinell hardness.

**SHEAR TEST:-**

Place the shear test attachment on the lower table, this attachment consists of cutter. The specimen is inserted in roles of shear test attachment & lift the lower table so that the zero is adjusted, then apply the load such that the specimen breaks in two or three pieces. If the specimen breaks in two pieces then it will be in angle shear, & if it breaks in three pieces then it will be in double shear.

**STUDY OF EXTENSOMETER:-**

This instrument is an attachment to Universal / Tensile Testing Machines. This measures the elongation of a test place on load for the set gauge length. The least count of measurement being 0.01 mm, and maximum elongation measurement up to 3 mm. This elongation measurement helps in finding out the proof stress at the required percentage elongation.

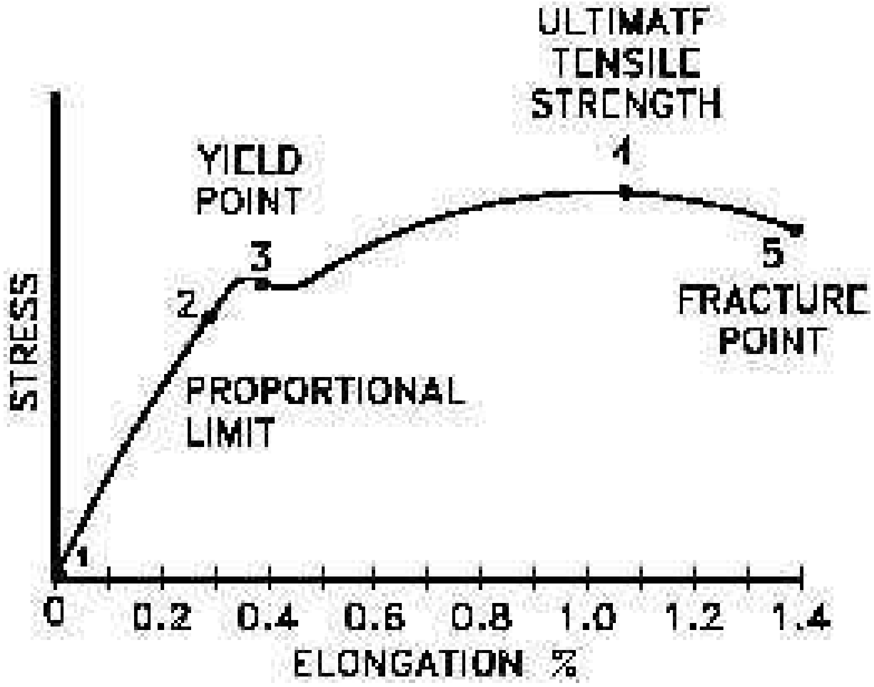
**WORKING OF THE INSTRUMENT:-**The required gauge length (between 30to 120 ) is set by adjusting the upper knife edges ( 3 ) A scale ( 2 ) is provided for this purpose . Hold the specimen in the upper and lower jaws of Tensile / Universal Testing Machine. Position the extensometer on the specimen. Position upper clamp (4) To press upper knife edges on the specimen. The extensometer will be now fixed to the specimen by spring pressure. Set zero on both the dial gauges by zero adjust screws (7 ). Start loading the specimen and take the reading of load on the machine at required elongation or the elongation at required load. Force setter accuracies mean of both the dial gauge ( 8) readings should be taken as elongation. It is very important to note & follow the practice of removing the extensometer from the specimen before the specimen breaks otherwise the instrument will be totally damaged. As a safety, while testing the instrument may be kept hanging from a fixed support by a slightly loose thread.

**TECHNICAL DATA:-**

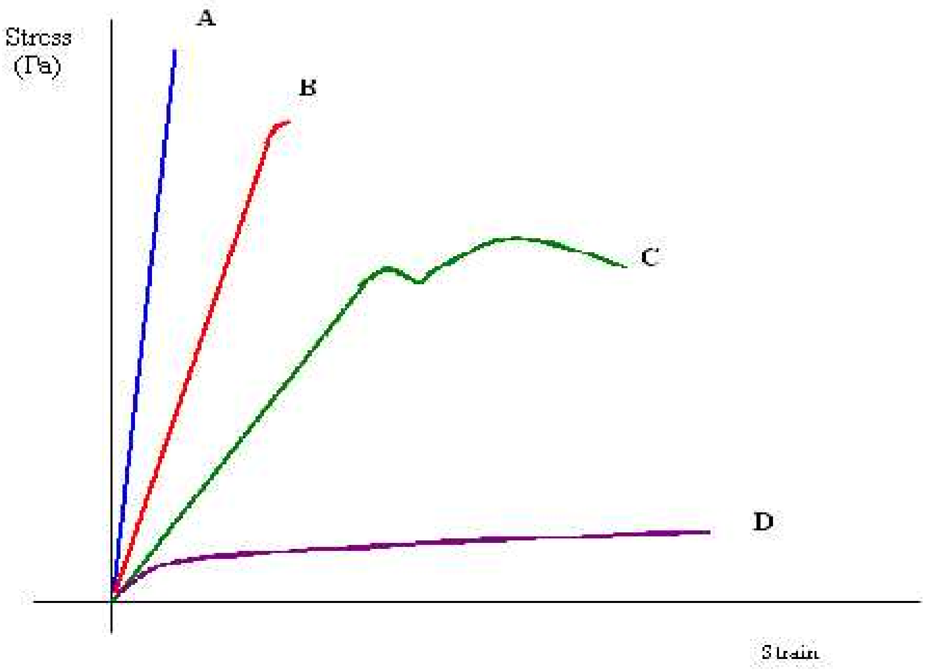
Measuring Range: 0 – 3 mm. Least Count: 0. 01 mm.
Gauge Length adjustable from: 30 – 120 mm

Specimen Size: 1 to 20mm Round or Flats up to 20 x 20 mm.

**A) Stress-strain graph of Mild Steel**



**B) Stress-strain graphs of different materials.**



- Curve **A** shows a **brittle** material. This material is also strong because there is little strain for a high stress. The fracture of a brittle material is sudden and catastrophic, with little or no plastic deformation. Brittle materials crack under tension and the stress increases around the cracks. Cracks propagate less under compression.
- Curve **B** is a **strong** material which is not ductile. Steel wires stretch very little, and break suddenly. There can be a lot of elastic strain energy in a steel wire under tension and it will "whiplash" if it breaks. The ends are razor sharp and such a failure is very dangerous indeed.
- Curve **C** is a **ductile** material
- Curve **D** is a **plastic** material. Notice a very large strain for a small stress.

------------------------------------------------------------------------------------------------------------------

**Experiment No. 8: -To determine tensile test on a metal.**

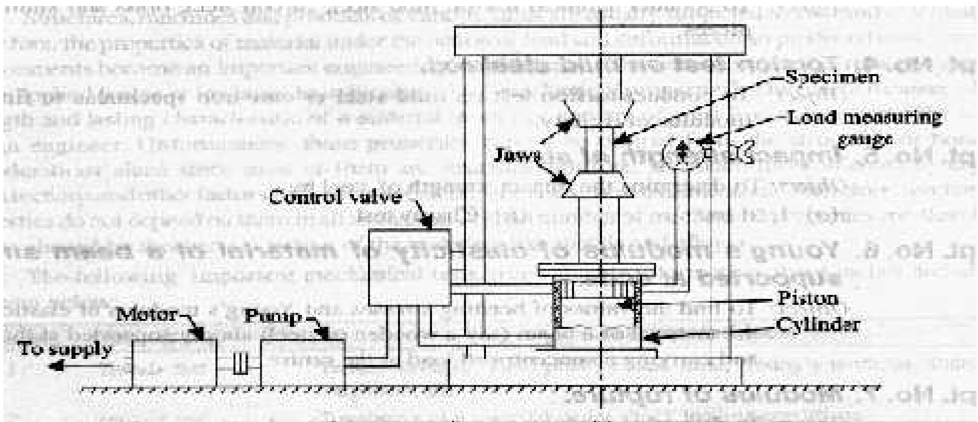
**Aim: -**To conduct a tensile test on a mild steel specimen and determine the following:

(i) Limit of proportionality (ii) Elastic limit
(iii) Yield strength (IV) Ultimate strength
(v) Young's modulus of elasticity (VI) Percentage elongation
(vii) Percentage reduction in area.

**APPARATUS: -**

(i) Universal Testing Machine (UTM)
(ii) Mild steel specimens
(iii) Graph paper
(iv) Scale
(v) Vernier Caliper

**DIAGRAM:-**



• Fig. 1. Tenstile testing machine

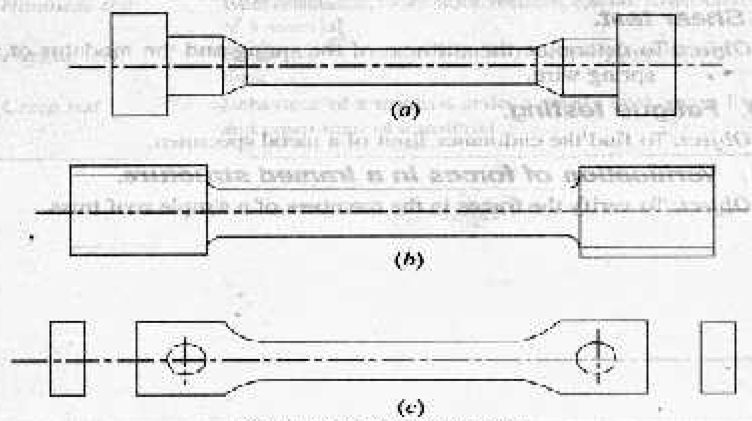
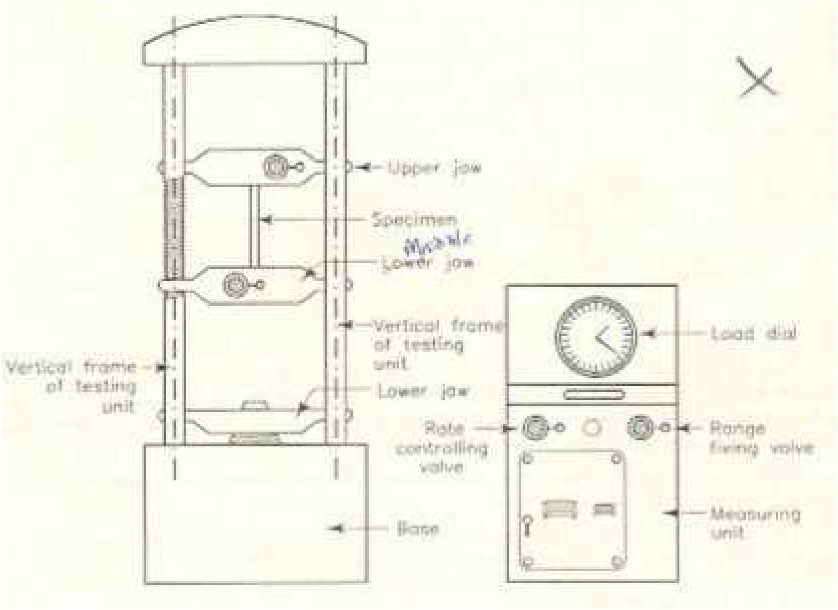


Fig. 2. Mild steel specimens.

**THEORY:-**The tensile test is most applied one, of all mechanical tests. In this test ends of test piece are fixed into grips connected to a straining device and to a load measuring device.

elastically deformed solid will return to its original form as soon as load is removed. However, if the load is too large, the material can be deformed permanently. The initial part of the tension curve which is recoverable immediately after unloading is termed. As elastic and the rest of the curve which represents the manner in which solid undergoes plastic deformation is termed plastic.

The stress below which the deformations essentially entirely elastic is known as the yield strength of material. In some material the onset of plastic deformation is denoted by a sudden drop in load indicating both an upper and a lower yield point. However, some materials do not exhibit a sharp yield point. During plastic deformation, at larger extensions strain hardening cannot compensate for the decrease in section and thus the load passes through a maximum and then begins to decrease. This stage the "ultimate strength"' which is defined as the ratio of the load on the specimen to original cross-sectional area, reaches a maximum value. Further loading will eventually cause 'neck' formation and rupture.



**PROCEDURE:-**

1) Measure the original length and diameter of the specimen. The length may either be length of gauge section which is marked on the specimen with a preset punch or the total length of the specimen.

2. Insert the specimen into grips of the test machine and attach strain-measuring device to it.

3. Begin the load application and record load versus elongation data.

4. Take readings more frequently as yield point is approached.

5. Measure elongation values with the help of dividers and a ruler.

6. Continue the test till Fracture occurs.

7. By joining the two broken halves of the specimen together, measure the final length and

**OBESERVATION:- 1. Material:**

A) Original dimensions
Length = ------------
Diameter = ---------
Area = --------------

B) Final Dimensions:

Length = ------------
Diameter = ---------
Area = --------------

**OBESERVATION TABLE:-**

| S.No | Load(N) | Original<br>Gauge length | Extension<br>(mm) | $\text{stress} = \frac{lc}{a}$<br>N/MM$^2$ | $\text{strain} = \frac{\text{increase in length}}{\text{original length}}$ |
|---|---|---|---|---|---|
| **1** | | | | | |
| **2** | | | | | |
| **3** | | | | | |
| **4** | | | | | |
| **5** | | | | | |

To plot the stress strain curve and determine the following.

(i) **Limit of proportion**

$$= \frac{lc \quad a \; li \quad o \; p}{o \quad a \quad o \; c \quad s}$$

**(ii) Elastic limit** = $\frac{\text{load at elastic limit}}{\text{Original area of c/s}}$ N/mm$^2$

**(iii) Yield strength =** $\frac{\text{Yield load}}{\text{Original area of cross-section}}$ N/mm$^2$

**(iv) Ultimate strength =** Maximum tensile load N/mm$^2$

Original area of cross-section

**(v) Young's modulus, E** = $\dfrac{\text{stress below propornality limit}}{\text{Corresponding strain}}$ $N/mm^2$

**(vi) Percentage elongation =** $\dfrac{\textbf{Final length (at fracture) – original length}}{\textbf{Original length}}$ **=…….%**

**(vii) Percentage reduction in area** = $\dfrac{\text{Original area-area at fracture}}{\text{Original area}}$ **=…….%**

**RESULT: -** i) Average Breaking Stress =
ii) Ultimate Stress =
iii) Average % Elongation=

**PRECAUTION:-**

1. If the strain measuring device is an extensometer it should be removed before necking begins.
2. Measure deflection on scale accurately & carefully

……………………………………………………………………………………………………………

**Experiment No. 9: To determine Compression test on a metal.**

**Aim:** To find the compressive strength of given specimen.

**Material and Equipment**

Universal testing machine,
Compression pads,
Given specimen,

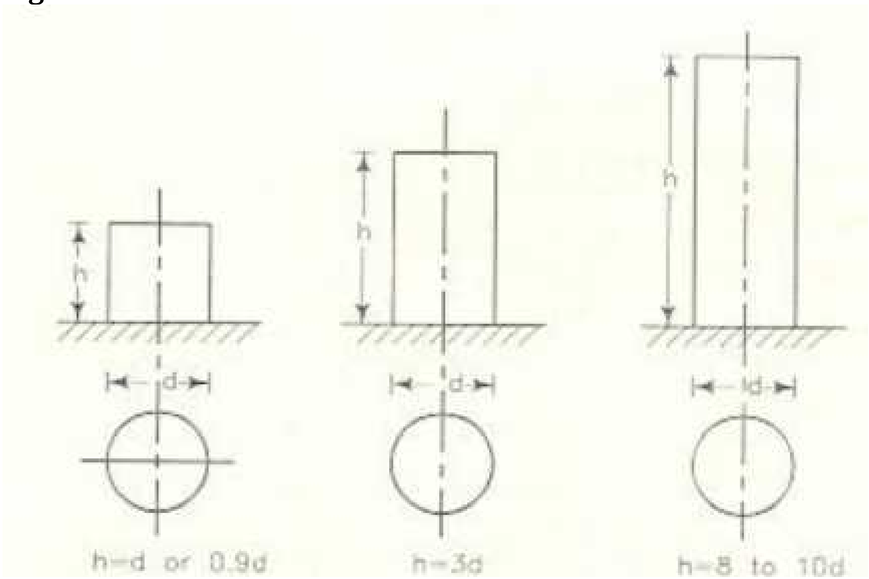
**Theory**

This is the test to know strength of a material under compression. Generally compression test is carried out to know either simple compression characteristics of material or column action of structural members. It has been observed that for varying height of member, keeping cross-sectional and the load applied constant, there is an increased tendency towards bending of a member.

Member under compression usually bends along minor axis, i.e, along least lateral dimension. According to column theory slenderness ratio has more functional value. If this ratio goes on increasing, axial compressive stress goes on decreasing and member buckles more and more. End conditions at the time of test have a pronounced effect on compressive strength of materials. Effective length must be taken according to end conditions assumed, at the time of the test.

As the ends of the member is made plain and fit between two jaws of the machine, fixed end is assumed for calculation of effective length. Effective length is taken as 0.5 L where L is actual length of a specimen.

**Figure**



**Observation**

Cross sectional area of the specimen perpendicular to the load=A=……mm $^2$
Load taken by the specimen at the time of failure, W=. …… (N)
Strength of the pin against shearing (s) = [W/A] N/mm $^2$

**Procedure**

1. Place the specimen in position between the compression pads.
2. Switch on the UTM
3. Bring the drag indicator in contact with the main indicator.
4. Select the suitable range of loads and space the corresponding weight in the pendulum and balance it if necessary with the help of small balancing weights
5. Operate (push) the button for driving the motor to drive the pump.
6. Gradually move the head control ever in left hand direction till the specimen fails.
7. Note down the load at which the specimen shears
8. Stop the machine and remove the specimen.
9. Repeat the experiment with other specimens.

**Precautions**

1. Place the specimen at centre of compression pads.
2. Stop the UTM as soon as the specimen fails.
3. Cross sectional area of specimen for compression test should be kept large as compared to the specimen for tension test: to obtain the proper degree of stability.

**Result**

Compressive strength of the specimen ………N/mm$^2$