

DEPARTMENT OF MECHANICAL ENGINEERING

Vision of the Department:

To become a Centre of excellence in the field of Mechanical Engineering by producing graduates with technical knowledge, research, consultancy and entrepreneurial skills along with leadership qualities, ethics and lifelong learning to cater the needs of the society

Mission of the Department:

- ✓ To impart quality education and training to nurture globally competitive mechanical engineers by effective teaching-learning practices and state-of-the art laboratories through eminent faculty.
- ✓ To establish linkages with industries and research organizations to bring excellence in problem solving skills, research and consultancy services.
- ✓ To empower the graduates with creative thinking, leadership qualities, lifelong learning skills, spirit of entrepreneurship, social and ethical values by offering value based education.
- ✓ To establish linkages with premier industries and research organizations to bring excellence in technical problem solving skills, research and consultancy services.
- ✓ To empower the graduates globally competent with creative thinking, leadership skills, lifelong learning, spirit of entrepreneurship, social and ethical values by offering value based education.

Course Outcomes

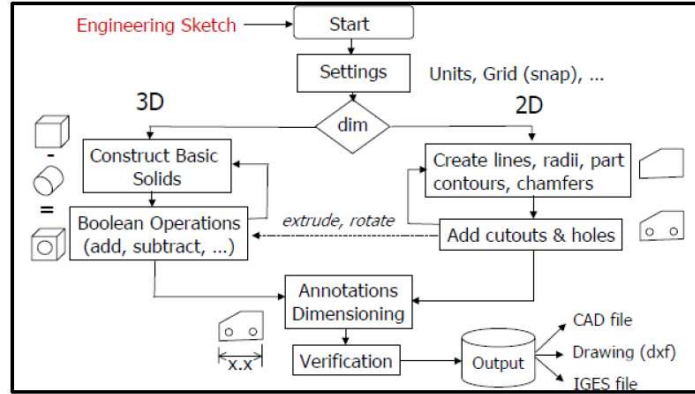
At the end of course the students will be able to

- Generate the CAD models.
- Write CNC programs for various machining operations.

Introduction to CAD/CAM LAB

Part A-CAD

Generic CAD Process



Introduction to Geometric transformation

- Essentially, computer graphics is concerned with generating, presenting and manipulating models of an object and its different views using computer hardware, software and graphic devices.
- Usually the numerical data generated by a computer at very high speeds is hard to interpret unless one represents the data in graphic format and it is even better if the graphic can be manipulated to be viewed from different sides, enlarged or reduced in size.
- Geometric transformation is one of the basic techniques that are used to accomplish these graphic functions involving scale change, translation to another location or rotating it by a certain angle to get a better view of it and can be handled conveniently using matrix algebra

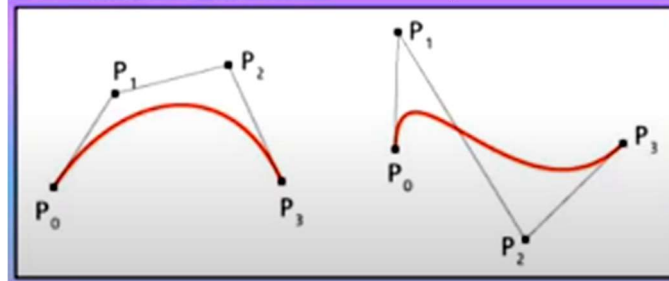
Three dimensional transformations- Translation, Scaling and Rotation

| | | |
|---|--|---|
| <p>1. Translation</p> $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix} \text{ or } \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$ | | <p>$P^* = [T]P$ Where [T] is transformation matrix</p> |
| <p>2. Scaling</p> $\begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$ | | |
| <p>3. Rotation (clockwise)</p> $\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$ | | |
| <p>4. Rotation (anti-clock)</p> $\begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$ | | |

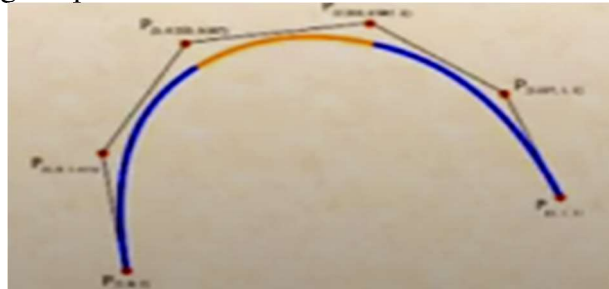
Some of the possible geometric transformations

Generating spline Bezier and B-spline

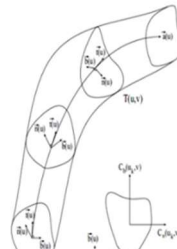

- Spline means a flexible strip used to produce a smooth curve through a designated set of points. A Spline curve specify by a giving set of coordinate positions called “Control Point”
- Bezier curves are defined using four control points, known as knots. Two of these are the end points of the curve, while the other two points control the shape of the curve.



- The degree of B-spline polynomial is independent of the number of vertices of the polygon, curve lies within the convex hull of its defining polygon. B-spline allows local control over the curve surface because each vertex affects the shape of the curve only over a range of parameter values where its associated basic function is non-zero

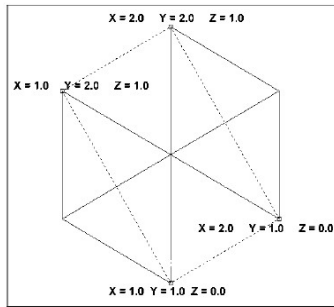


Introduction to sweep surfaces and Surface of revolution

| <h3>Swept Surfaces (Generalized Cylinder)</h3> <th data-bbox="767 1317 1407 1720"> <h3>Surface of revolution</h3> </th> | <h3>Surface of revolution</h3> | | | | | | | | | | | | | |
|--|---|--|---|--|--|--|--|--|--|--|--|--|--|---|
| <p>✓ Swept surfaces are generated by moving a 2D curve along a trajectory in 3D space.</p> <p>✓ Curve can also change its shape and orientation during sweeping.</p> <p>✓ Generalized cylinder is the shape generated when a 2D contour is swept along a 3D trajectory.</p> <p>✓ Contour define the cross-section of the object.</p> <p>✓ Trajectory is the axis of the object.</p>  | <p>✓ It's generated by a 2D contour swept along a circular trajectory</p> <p>✓ Equivalently the solid can be generated by rotation of the 2D contour around an axis.</p> <p>✓ If the base curve isn't closed (usual case), it's always possible generate a solid by adding two circular planar surface for the bottom and the top of the solid.</p>  | | | | | | | | | | | | | |
| <h3>Sweep Line</h3> <table border="1"> <tr> <td> Two Limits Inputs: a. Guide curve 1 b. Guide Curve 2 </td> <td> Limit and middle Inputs: a. Guide curve 1 b. Guide Curve 2(Middle Curve) </td> <td> With reference surface Inputs: a. Guide curve 1 b. Reference Surface c. Angle </td> </tr> <tr> <td> With reference Curve Inputs: a. Guide curve 1 b. Reference Curve c. Angle </td> <td> With tangency surface Inputs: a. Guide curve 1 b. Tangency Surface </td> <td> With draft direction Inputs: a. Guide curve 1 b. Draft Direction c. Angle </td> </tr> <tr> <td colspan="3"> With 2 tangency surfaces Inputs: a. Spine b. 1st tangency surface c. 2nd tangency surface </td> </tr> </table> | Two Limits Inputs: a. Guide curve 1 b. Guide Curve 2 | Limit and middle Inputs: a. Guide curve 1 b. Guide Curve 2(Middle Curve) | With reference surface Inputs: a. Guide curve 1 b. Reference Surface c. Angle | With reference Curve Inputs: a. Guide curve 1 b. Reference Curve c. Angle | With tangency surface Inputs: a. Guide curve 1 b. Tangency Surface | With draft direction Inputs: a. Guide curve 1 b. Draft Direction c. Angle | With 2 tangency surfaces Inputs: a. Spine b. 1 st tangency surface c. 2 nd tangency surface | | | <h3>Sweep Conic</h3> <table border="1"> <tr> <td> Two guide curves Input: a. Guide curve 1. b. Tangency (Surface/plane). c. Last guide curve d. Tangency (Surface/plane) </td> <td> Three guide curves Input: a. Guide curve 1. b. Tangency (Surface/plane). c. Guide curve 2 d. Last Guide curve e. Tangency (Surface/plane) </td> <td> Four guide curves Input: a. Guide curve 1. b. Tangency (Surface/plane). c. Guide curve 2 d. Guide curve 3 e. Last Guide curve </td> <td> Five guide curves Input: a. Guide curve 1 b. Guide curve 2 c. Guide curve 3 d. Guide curve 4 e. Last Guide curve </td> </tr> </table> | Two guide curves Input: a. Guide curve 1. b. Tangency (Surface/plane). c. Last guide curve d. Tangency (Surface/plane) | Three guide curves Input: a. Guide curve 1. b. Tangency (Surface/plane). c. Guide curve 2 d. Last Guide curve e. Tangency (Surface/plane) | Four guide curves Input: a. Guide curve 1. b. Tangency (Surface/plane). c. Guide curve 2 d. Guide curve 3 e. Last Guide curve | Five guide curves Input: a. Guide curve 1 b. Guide curve 2 c. Guide curve 3 d. Guide curve 4 e. Last Guide curve |
| Two Limits Inputs: a. Guide curve 1 b. Guide Curve 2 | Limit and middle Inputs: a. Guide curve 1 b. Guide Curve 2(Middle Curve) | With reference surface Inputs: a. Guide curve 1 b. Reference Surface c. Angle | | | | | | | | | | | | |
| With reference Curve Inputs: a. Guide curve 1 b. Reference Curve c. Angle | With tangency surface Inputs: a. Guide curve 1 b. Tangency Surface | With draft direction Inputs: a. Guide curve 1 b. Draft Direction c. Angle | | | | | | | | | | | | |
| With 2 tangency surfaces Inputs: a. Spine b. 1 st tangency surface c. 2 nd tangency surface | | | | | | | | | | | | | | |
| Two guide curves Input: a. Guide curve 1. b. Tangency (Surface/plane). c. Last guide curve d. Tangency (Surface/plane) | Three guide curves Input: a. Guide curve 1. b. Tangency (Surface/plane). c. Guide curve 2 d. Last Guide curve e. Tangency (Surface/plane) | Four guide curves Input: a. Guide curve 1. b. Tangency (Surface/plane). c. Guide curve 2 d. Guide curve 3 e. Last Guide curve | Five guide curves Input: a. Guide curve 1 b. Guide curve 2 c. Guide curve 3 d. Guide curve 4 e. Last Guide curve | | | | | | | | | | | |

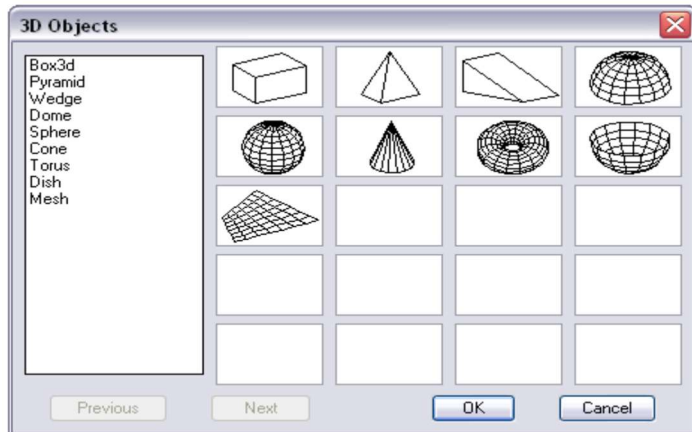
Create wireframe, surface and solid models.

A wireframe model is a skeletal description of a 3D object. There are no surfaces in a wireframe model; it consists only of points, lines, and curves that describe the edges of the object. With AutoCAD you can create wireframe models by positioning 2D (planar) objects anywhere in 3D space. AutoCAD also provides some 3D wireframe objects, such as 3D polylines (that can only have a CONTINUOUS line type) and splines. Because each object that makes up a wireframe model must be independently drawn and positioned, this type of modelling can be the most time-consuming.

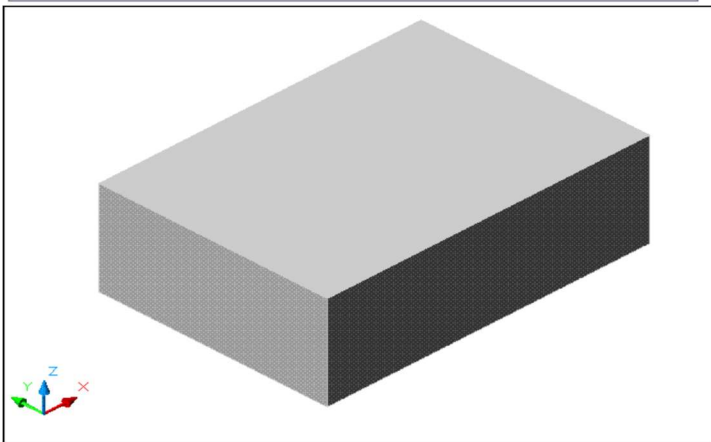


Wireframe model

Surface modeling is more sophisticated than wireframe modeling in that it defines not only the edges of a 3D object, but also its surfaces. The AutoCAD surface modeler defines faceted surfaces using a polygonal mesh. Because the faces of the mesh are planar, the mesh can only approximate curved surfaces. With Mechanical Desktop, you can create true curved surfaces. To differentiate these two types of surfaces, AutoCAD calls faceted surfaces, meshes.

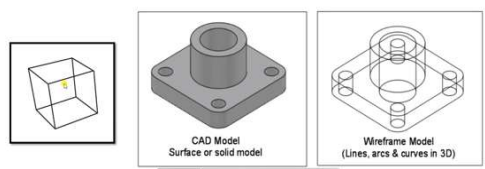


Solid modeling is the easiest type of 3D modeling to use. With the AutoCAD solid modeler, you can make 3D objects by creating basic 3D shapes: boxes, cones, cylinders, spheres, wedges, and tori (donuts). You can then combine these shapes to create more complex solids by joining or subtracting them or finding their intersecting (overlapping) volume. You can also create solids by sweeping a 2D object along a path or revolving it about an axis.

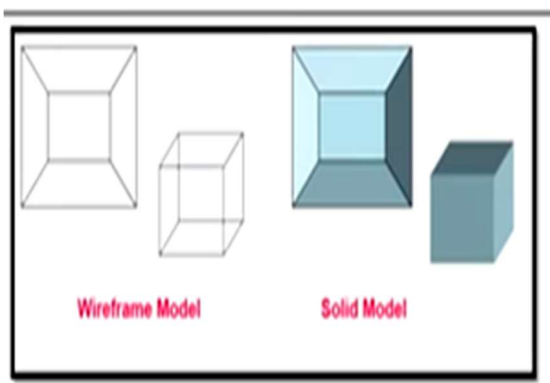
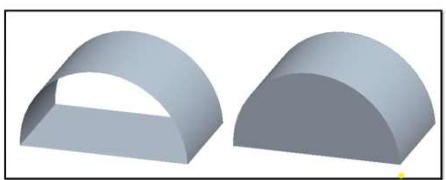


NOTE: Because each modeling type uses a different method for constructing 3D models and editing methods vary in their effect on the different model types, it is recommended that you not mix modelling methods.

Wireframe Modeling




Surface Modeling



Detailed Procedure to use some important Commands is explained below:

1. **LINE:** Line Command is used to draw a chain of straight Lines.

- Select the “Line Chain” in the **Line** command.
- Move the cursor and click on the starting point of the line and again move the cursor to the end point of the line and click the mouse.
- A line is generated with dimension.
- Clicking the scroll button of the mouse once will terminate the current chain of the lines and a fresh chain can be drawn as required.
- To come out of the line command click the scroll button twice or click on the “Select, ” button in the Menu Bar.
- Double click on the dimension and enter the required value.

2. **RECTANGLE:** A *Rectangle* can be drawn in many ways. Two of them, viz “Corner” and “Center” are explained below.

Corner Rectangle:

- Select the “Corner Rectangle” option in the “Rectangle” command.
- Move the cursor to one corner of the required rectangle and click and select the diagonally opposite corner with the cursor to complete the rectangle.
- Double-click the scroll button to come out of the “Rectangle” command.
- Double click on each dimension and enter the required values.

Center Rectangle:

- Select the “Center Rectangle” option in the “Rectangle” command.
- Move the cursor to the centre of the required rectangle and click.
- Select any of the corners the rectangle with the cursor to complete it.
- Edit the dimensions as required

3. **CIRCLE:** A Circle can be drawn in many ways.

One of them, viz “Center and Point” is explained below.

“Center and Point” Circle:

- Select “Center and Point” option in the “Circle” command.
- Move the cursor to the center point of the required circle and click.
- Select another point on the circumference to complete the circle.
- Double-click the scroll button to come out of the command.
- Double click on the dimensions and enter the required values.

4. **ARC:** An *Arc* can be drawn in many ways.

Two of them, viz “3-Point” and “Center and Ends” are explained below.

“3-Point” Arc:

- Select the “3-Point” option in the “Arc” command.
- Move the cursor to the start point of the required arc and click.
- Now move the cursor to the end point of the arc and click.
- Select a point on the desired side of the arc and click to complete the arc.
- Edit the dimensions as required.

“Center and Ends” Arc:

- Select the “Center and Ends” option in the “Arc” command.
- Move the cursor to the centre point of the arc and click.
- Move the start and end points in the desired direction to complete the arc.
- Edit the dimensions as required.

5. **ELLIPSE:** An *Ellipse* can be drawn in two ways,
viz “Axis Ends” and “Center and Axis,” which are explained below:

“Axis Ends” Ellipse:

- Select the “*Axis Ends Ellipse*” option in the “Ellipse” command.
- Move the cursor and select the start and end points of any one axis of the ellipse.
- Select one end point of the other axis to complete the ellipse.
- Edit the dimensions as required.
- Move the cursor

“Center and Axis” Ellipse:

- Select the “*Center and Axis*” option in the “Ellipse” command.
- Move the cursor and select the centre point of the required ellipse.
- Select one end of the major axis.
- Select one end of the minor axis also to complete the Ellipse.
- Correct the dimensions as required.

6. **SPLINE:** A “Spline” is a smooth curved line passing through all the specified points.

- Select the “Spline” command.
- Select the points in the sequence along the path of the required spline.
- Double click the scroll button to come out of “Spline” command.
- Edit the point locations as required.

7. **FILLET:** A fillet is used to round off sharp edges by a smooth curve,
either circular or elliptical. The “Fillet” option can be with or without trimming.

The commands are explained below:

“Circular” Fillet:

- Select the ‘*Circular*’ option in the ‘Fillet’ command.
- Select the lines to be filleted.
- Edit the fillet radius as required.

“Circular Trim” Fillet:

- Select the “*Circular Trim*” option in the “Fillet” command.
- Select the lines to be filleted and edit the fillet radius.
- Notice that the construction lines forming the edge are trimmed.

“Elliptical” Fillet:

- Select the “*Elliptical*” option in the “Fillet” command.
- Select the lines to be filleted and edit the fillet radii as required.

“Elliptical Trim” Fillet:

- Select the “*Elliptical Trim*” option in the “Fillet” command.
- Select the lines to be filleted and edit the fillet radii.
- Notice that the construction lines forming the edge are trimmed.

8. **CHAMFER:** A chamfer is used to remove sharp edges by an angular corner.

In the drawing it can be shown with or without trimming.

Chamfer:

- Select the “*Chamfer*” option in the “Chamfer” command.
- Select the lines to be chamfered.
- Edit the chamfer parameters as required.

“Chamfer Trim”:

- Select the “*Chamfer Trim*” option in the “Chamfer” command.
- Select the lines to be chamfered and edit the chamfer parameters.
- Notice that the construction lines forming the edge are trimmed.

9. **TEXT:** A text command is used to write text, as explained below:
- Select the “*Text*” command in the menu bar.
 - Move the cursor to a point and click where the text is to be written.
 - Select another point vertically above the first point.
 - The “text window” pops up where we can enter the required text, and select Text symbol (Font and Position) and click OK in the dialogue box.
 - Edit the text parameters.
10. **OFFSET:** An offset command is used to create another entity by offsetting the existing entity. The commands are explained below:
- Select the “*Offset*” command in the menu bar.
 - Select the existing entity (say a line) to be offset, now an arrow appears showing the direction of offset. Enter the distance in window bar. If we want to change the direction, enter negative value and click green tick mark, now offset image is created.
11. **CENTER LINE:** A center line is used to create a construction line which is useful in revolving / mirroring an object. The command is explained below:
- Select the “*Centerline*” command in the menu bar.
 - Select two points each along ‘x’ and ‘y’ axes to create centerlines.
 - “*Centerline*” can also be created at any other location and at any angle.
12. **POINT:** ‘*Point*’ command is used to create a construction “*Point*’(s).”
- Select the “*Point*” command in the menu bar.
 - Select *Point*(s), anywhere on the sketching plane and at the required location.
 - The points can be used to draw different entities.
13. **COORDINATE SYSTEM:** “*Coordinate system*” command is used to create
- Select the “*Coordinate system*” command in the menu bar.
 - Select *Point*(s) where another *Coordinate system*(s) is required.
 - Entities can now be drawn w.r.t. the new coordinate system.
14. **PALETTE:** “*Palette*” is a command by which we can import data from *sketcher Palette* into the drawing. By this command we can import documents, polygons, profiles, shapes and stars etc. The command is explained below:
- Select the “*Palette*” command in the menu bar.
 - Now *sketcher Palette* window pops up and we can import documents, polygons, profiles, shapes and stars etc. by selecting and dragging them into the drawing.
15. **MODIFY:** “*Modify*” is a command by which we can modify dimensions or the shape of objects. The command is explained below:
- Select the “*Modify*” command in the menu bar.
 - Select the dimensions to be modified, now window pops up and we can enter the required dimension and click on the tick mark.
 - Similarly we can also modify the shape of the object by selecting the node points and drag it to the required shape.
16. **MIRROR:** “*Mirror*” is a command by which we can create a mirror image of the object.
- Select the “*Mirror*” command in the menu bar.
 - Select the object, now the mirror command gets activated, click on the *Mirror* command and click on the center line to get the mirror image of the object.
17. **DIVIDE:** “*Divide*” is a command by which we can divide an entity at the selected point(s). The command is explained below:
- Select the “*Divide*” command in the menu bar.
 - Select the points on the entity, and the entity gets divided at the selected *Points*.

18. **DELETE SEGMENT:** “*Delete segment*” is a command by which we can delete an entity. The command is explained below:
- Select the “*Delete segment*” command in the menu bar.
 - Select an entity(s) to deleted, and the entity gets deleted.
19. **CORNER:** With “*Corner*” command we can join the entities to form a corner.
- Select the “*Corner*” command in the menu bar.
 - Select two entities, now the two entities get joined in to a corner.
20. **ROTATE RESIZE:** With “*Rotate resize*” command we can rotate / resize entities.
- Select the “*Rotate resize*” command in the menu bar.
 - Select the entity, now a rotate symbol is visible (on top right corner), and scaling arrows visible at the bottom right corner, by which we can rotate / resize the selected entity.
21. **VERTICAL:** With “*Vertical*” command we can make any entity ‘vertical’
- Select the “*Vertical*” command in the menu bar.
 - Select any inclined line, the line becomes vertical.
22. **HORIZONTAL:** With “*Horizontal*” command we can make an inclined line ‘horizontal’
- Select the “*Horizontal*” command in the menu bar.
 - Select any inclined line, the line becomes ‘horizontal’.
23. **PERPENDICULAR:** With “*Perpendicular*” command we can make the entities perpendicular to each other as explained below:
- Select the “*Perpendicular*” command in the menu bar.
 - Select the first entity and select then the second entity, the second entity gets perpendicular to the first entity.
24. **TANGENT:** With “*Tangent*” command we can make the entity tangent to another entity
- Select the ‘*Tangent*’ command in the menu bar.
 - Select a line and then select a point on the periphery of a circle or a curve, the line becomes tangent to the selected curve.
25. **MID-POINT:** With “*Mid-point*” command we can create construction midpoint on the
- Select the “*Mid-point*” command in the menu bar.
 - Now select the line, the construction ‘midpoint’ is created on the line.
26. **COINCIDENT:** With “*Coincident*” command, two entities can be made coincident
- Select the “*Coincident*” command in the menu bar.
 - Select the first entity and select then the second entity, the second entity gets coincident to the first entity.
27. **SYMMETRIC:** With “*Symmetric*” command, a line can be made symmetric to horizontal or vertical axes as explained below:
- Select the “*Symmetric*” command in the menu bar.
 - If a line is required to be made symmetric to x or y axis, Select the end points of the line and click on the axis required, now the line becomes symmetrical to the selected axis.
28. **EQUAL:** With “*Equal*” command, two entities can be made equal as explained below:
- Select the “*Equal*” command in the menu bar.
 - Select the first entity and select then the second entity, both the entities become equal to each other.
29. **PARALLEL:** With “*Parallel*” command two entities can be made parallel to each other
- Select the “*Parallel*” command in the menu bar.
 - Select the first entity and select then the second entity, the second entity gets parallel to the first entity.
30. **NORMAL:** With “*Normal*” command we can find out the distance between the two
- Select the “*Normal*” command in the menu bar.
 - Select any two points and click on the side where the dimension is required, the dimension appears and can be verified /edited.

3D Transformation C Program

Program No: 1

Date:

Aim : To write a program for performing 3D concatenations to SCALE a tetrahedron PQRS about point S by uniformly by 2 times

```
3D Transformation C Program
File Edit Search Run Compile Debug Project Options Window Help
S-3DSCAL.C 3-[+]
#include<stdio.h>
#include<conio.h>
#include<math.h>
//program for performing 3D concatenations
//to scale a tetrahedronPQRS about point S by uniformly by 2 times
void main()
{
    int m=4, n=4, p, q, c, d, k, h,ns,ch,i,j,r,rf;
    float th;
    float first[10][10], second[10][10], mult1[10][10],mult2[10][10],mult3[10][10];
    float tx,ty,sh;
    float trans[4][4],trans2[4][4],scale[4][4],mult[10][10],sum2=0,sum3=0;
    clrscr();
//to create object matrix
    printf("Enter coordinates of object matrix\n");

    for (d = 0; d < n; d++)
    {
        for (c = 0; c < m-1; c++)
        {
            scanf("%f", &first[c][d]);
            first[m-1][d]=1;
        }
    }
    for (c = 0; c < m; c++)
    {
        for (d = 0; d < n; d++)
            printf("%f\t", first[c][d]);
        printf("\n");
    }
//back translation matrix
    printf("\n");
    for(i=0;i<4;i++)
    {
        for(j=0;j<4;j++)
        {
            if(i==j)
                trans[i][j]=1;
            else
                trans[i][j]=0;
        }
    }
}
```

```

trans[0][3]=-1*first[0][3];
trans[1][3]=-1*first[1][3];
trans[2][3]=-1*first[2][3];
printf("%f\t",trans[i][j]);
}
printf("\n");
}
for (i = 0; i < 4; i++) {
for (j = 0; j < n; j++) {
for (k = 0; k < 4; k++) {
    sum = sum + trans[i][k]*first[k][j];
}
mult1[i][j] = sum;
sum = 0;
}
}
//scaling matrix
printf("\n");
sh=2;
for(i=0;i<4;i++)
{
for(j=0;j<4;j++)
{
if(i==j)
    scale[i][j]=sh;
else
    scale[i][j]=0;

scale[3][3]=1;
printf("%f\t",scale[i][j]);
}
printf("\n");
}
for (i = 0; i < 4; i++) {
for (j = 0; j < n; j++) {
for (k = 0; k < 4; k++) {
    sum2 = sum2 + scale[i][k]*mult1[k][j];
}
mult2[i][j] = sum2;
sum2 = 0;
}
}
//forward translation
printf("\n");
for(i=0;i<4;i++)
{
for(j=0;j<4;j++)
{
if(i==j)
    trans2[i][j]=1;
else
    trans2[i][j]=0;
}
}

```

```

trans2[0][3]=first[0][3];
trans2[1][3]=first[1][3];
trans2[2][3]=first[2][3];
printf("%f\t",trans2[i][j]);
}
printf("\n");
}
for (i = 0; i < 4; i++) {
for (j = 0; j < n; j++) {
for (k = 0; k < 4; k++) {
sum3 = sum3 + trans2[i][k]*mult2[k][j];
}
mult3[i][j] = sum3;
sum3 = 0;
}
}
printf("New Transformed coordinates of object are:\n");
for (i = 0; i < 4; i++)
{
for (j = 0; j < n; j++)
printf("%f\t", mult3[i][j]);
printf("\n");
}
getch();

```

Input

```

Linking
EXE file : ..\SOURCE\S-3DSCAL.EXE
Linking : \TURBOC3\LIB\CS.LIB

Total      Link
Lines compiled: 738  PASS 2
Warnings: 0        0
Errors: 0          0

Available memory: 1969K
Success :

Enter coordinates of object matrix
5 5 5 10 5 10 15 5 5 10 15 8 _

```

Output

```

5.000000    10.000000    15.000000    10.000000
5.000000    5.000000     5.000000    15.000000
5.000000    10.000000    5.000000     8.000000
1.000000    1.000000     1.000000     1.000000

1.000000    0.000000     0.000000    -10.000000
0.000000    1.000000     0.000000    -15.000000
0.000000    0.000000     1.000000    -8.000000
0.000000    0.000000     0.000000     1.000000

2.000000    0.000000     0.000000     0.000000
0.000000    2.000000     0.000000     0.000000
0.000000    0.000000     2.000000     0.000000
0.000000    0.000000     0.000000     1.000000

1.000000    0.000000     0.000000    10.000000
0.000000    1.000000     0.000000    15.000000
0.000000    0.000000     1.000000     8.000000
0.000000    0.000000     0.000000     1.000000
New Transformed coordinates of object are:
0.000000    10.000000    20.000000    10.000000
-5.000000   -5.000000    -5.000000    15.000000
2.000000    12.000000    2.000000     8.000000
1.000000    1.000000     1.000000     1.000000

```

Program No: 2**Date:**

Aim: To Write a program for performing 3D concatenations to **ROTATE** a tetrahedron PQRS about Z-axis by 30° keeping a point S fixed.

```
File Edit Search Run Compile Debug Project Options Window Help
D-3DROTA.C
#include<stdio.h>
#include<conio.h>
#include<math.h>
//program for performing 3D concatenations
//to rotate a tetrahedron PQRS about Z-axis by 30 deg keeping point S fixed
void main()
{
    int m=4, n=4, p, q, c, d, k, h,ns,ch,i,j,r,rf;
    float th;
    float first[10][10], second[10][10], mult1[10][10],mult2[10][10],mult3[10][10];
    float tx,ty;
    float trans[4][4],trans2[4][4],rot[4][4],mult[10][10],sum2=0,sum3=0;
    clrscr();
    //to create object matrix
    printf("Enter coordinates of object matrix\n");

    for (d = 0; d < n; d++)
    {
        for (c = 0; c < m-1; c++)
        {
            scanf("%f", &first[c][d]);
            first[m-1][d]=1;
        }
    }
    for (c = 0; c < m; c++)
    {
        for (d = 0; d < n; d++)
            printf("%f\t", first[c][d]);
        printf("\n");
    }
    //back translation matrix
    printf("\n");
    for(i=0;i<4;i++)
    {
        for(j=0;j<4;j++)
        {
            if(i==j)
                trans[i][j]=1;
            else
                trans[i][j]=0;
            trans[0][3]=-1*first[0][3];
            trans[1][3]=-1*first[1][3];
            trans[2][3]=-1*first[2][3];
            printf("%f\t",trans[i][j]);
        }
        printf("\n");
    }
    for (i = 0; i < 4; i++) {
        for (j = 0; j < n; j++) {
            for (k = 0; k < 4; k++) {
                sum = sum + trans[i][k]*first[k][j];
            }
            mult1[i][j] = sum;
            sum = 0;
        }
    }
}
```

```

//rotation matrix
printf("\n");
th=3.1415*30/180;
for(i=0;i<4;i++)
{
for(j=0;j<4;j++)
{
if(i==j)
rot1[i][j]=cos(th);
else
rot1[i][j]=0;

rot1[0][1]=-sin(th);
rot1[1][0]=sin(th);
rot1[2][2]=1;
rot1[3][3]=1;
printf("%f\t",rot1[i][j]);
}
printf("\n");
}
for (i = 0; i < 4; i++) {
for (j = 0; j < n; j++) {
for (k = 0; k < 4; k++) {
sum2 = sum2 + rot1[i][k]*mult1[k][j];
}
mult2[i][j] = sum2;
sum2 = 0;
}
}
//forward translation
printf("\n");
for(i=0;i<4;i++)
{
for(j=0;j<4;j++)
{
if(i==j)
trans2[i][j]=1;
else
trans2[i][j]=0;

trans2[0][3]=first[0][3];
trans2[1][3]=first[1][3];
trans2[2][3]=first[2][3];
printf("%f\t",trans2[i][j]);
}
printf("\n");
}
for (i = 0; i < 4; i++) {
for (j = 0; j < n; j++) {
for (k = 0; k < 4; k++) {
sum3 = sum3 + trans2[i][k]*mult2[k][j];
}
mult3[i][j] = sum3;
sum3 = 0;
}
}
printf("New Transformed coordinates of object are:\n");
for (i = 0; i < 4; i++)
{
for (j = 0; j < n; j++)
printf("%f\t", mult3[i][j]);
printf("\n");
}
getch();
}

```

Input

```
Compiling
Main file: D-3DROTA.C
Compiling: EDITOR -> D-3DROTA.C

          Total   File
Lines compiled: 742   742
Warnings: 0         0
Errors: 0          0

Available memory: 1969K
Success          :   Press any key
```

```
Enter coordinates of object matrix
5 5 5 10 5 10 15 5 5 10 15 8_
```

Output

```
5.000000    10.000000    15.000000    10.000000
5.000000     5.000000     5.000000    15.000000
5.000000    10.000000     5.000000     8.000000
1.000000     1.000000     1.000000     1.000000

1.000000     0.000000     0.000000    -10.000000
0.000000     1.000000     0.000000    -15.000000
0.000000     0.000000     1.000000     -8.000000
0.000000     0.000000     0.000000     1.000000

0.866033    -0.499987     0.000000     0.000000
0.499987     0.866033     0.000000     0.000000
0.000000     0.000000     1.000000     0.000000
0.000000     0.000000     0.000000     1.000000

1.000000     0.000000     0.000000    10.000000
0.000000     1.000000     0.000000    15.000000
0.000000     0.000000     1.000000     8.000000
0.000000     0.000000     0.000000     1.000000
New Transformed coordinates of object are:
10.669701    14.999866    19.330032    10.000000
3.839735     6.339668     8.839602    15.000000
5.000000    10.000000     5.000000     8.000000
1.000000     1.000000     1.000000     1.000000
```

Program No: 3

Date:

Aim : To Write a program to draw Bezier Curve

```
//Write a program to draw Bezier Curve
#include<stdio.h>
#include<graphics.h>
#include<math.h>
void main()
{
    int gd=DETECT,gm;
    int x[4],y[4],i;
    initgraph(&gd,&gm,"C:\\VC\\BGI");
    printf("Enter the x and y coordinates of the 4 coordinates");
    for(i=0;i<4;i++)
    {
        scanf("%d%d",&x[i],&y[i]);
    }
    bezier(x,y);
    getch();
    closegraph();
}
bezier(int x[4],int y[4])
{
    int i;
    double t;
    for(t=0.0;t<1.0;t+=0.0005)
    {
        double xt=pow(1-t,3)*x[0]+3*t*pow(1-t,2)*x[1]+3*(1-t)*pow(t,2)*x[2]+pow(t,3)*x[3];
        double yt=pow(1-t,3)*y[0]+3*t*pow(1-t,2)*y[1]+3*(1-t)*pow(t,2)*y[2]+pow(t,3)*y[3];
        putpixel(xt,yt,WHITE);
    }
    for(i=0;i<4;i++)
    {
        putpixel(x[i],y[i],YELLOW);
    }
}
```

Input

```
Compiling
Main file: BEZIERCU.C
Compiling: EDITOR + BEZIERCU.C

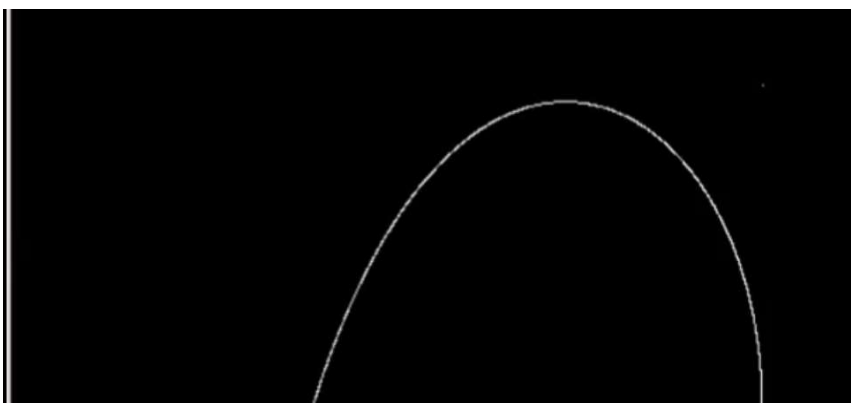
Total File
Lines compiled: 894 894
Warnings: 1 1
Errors: 0 0

Available memory: 1969K
Warnings :
```

```
Message
Compiling BEZIERCU.C:
Warning BEZIERCU.C 34: Function should return a value

Enter the x and y coordinates of the 4 coordinates200
400
300
100
500
200
500
500
400
```

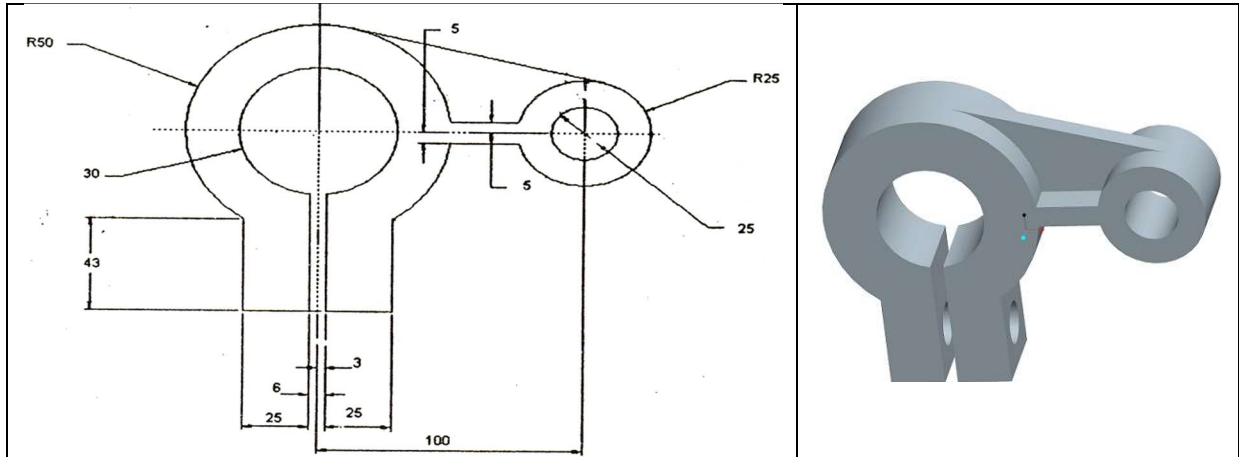
Output



3D- MODELING

Experiment No: 4

Date:



AIM: To draw the 3D model with given dimensions using CATIA software

Operating System: Windows XP/7/8/9/10

System configuration: 2 GB Ram, 100GB HD with Intel Processor

Operations:

CUT

- Insert → Extrude → Cut → Placement → define → select sketching plane.
Now you will enter into Sketch Mode.
- Draw an entity at the required position with required dimension Click on ✓ → Okay → Thru all → done → Ok (Model Dialog Box) GO to View → default Orientation
Now you can see a 3-D model (With the required cut)

RIP

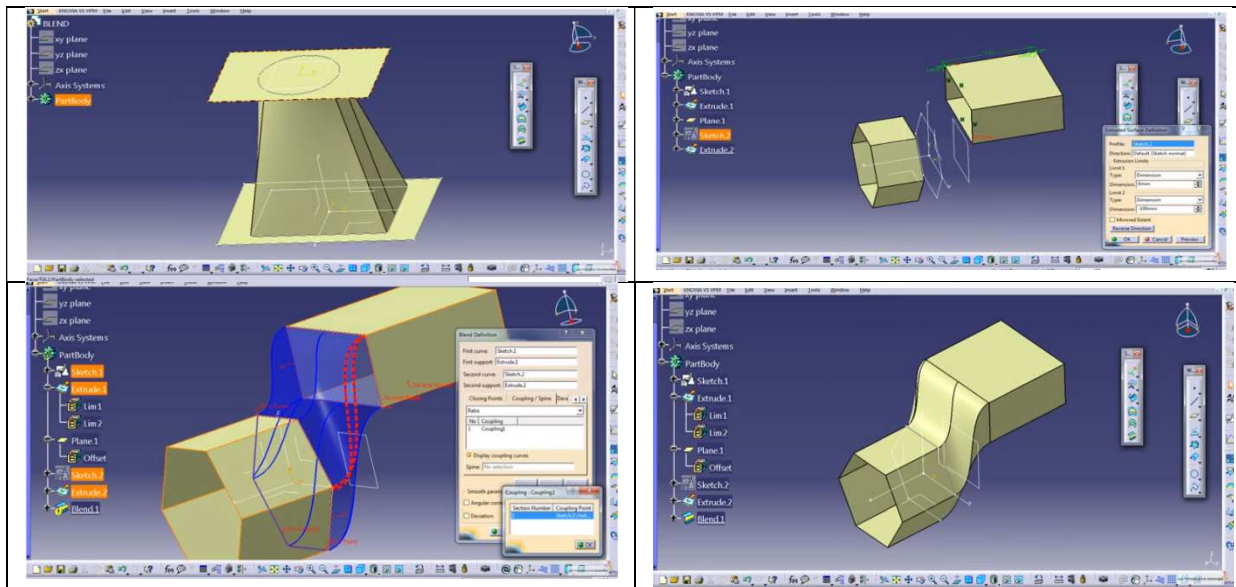
- Insert → Extrude → Cut → Placement → define → select sketching plane.
Now you will enter into Sketch Mode.

Draw an entity at the required position

Experiment No: 5

Date:

Aim: To blend surfaces with CATIA



Blend surfaces using Catia

Procedure:

1. Open file and save as Blend, Select X-Y plane
2. Select Shape under start choose Geometrical Shape design (Create 1st and 2nd Curves and Two supports)
3. Generate a Circle with 25mm radius, Blend part body is ready
4. Go to Start-Mechanical Design-Part design, give the name Blend1
5. Select offset plane, Extrude-Sketch with point-1 and point-2- Hide and show.

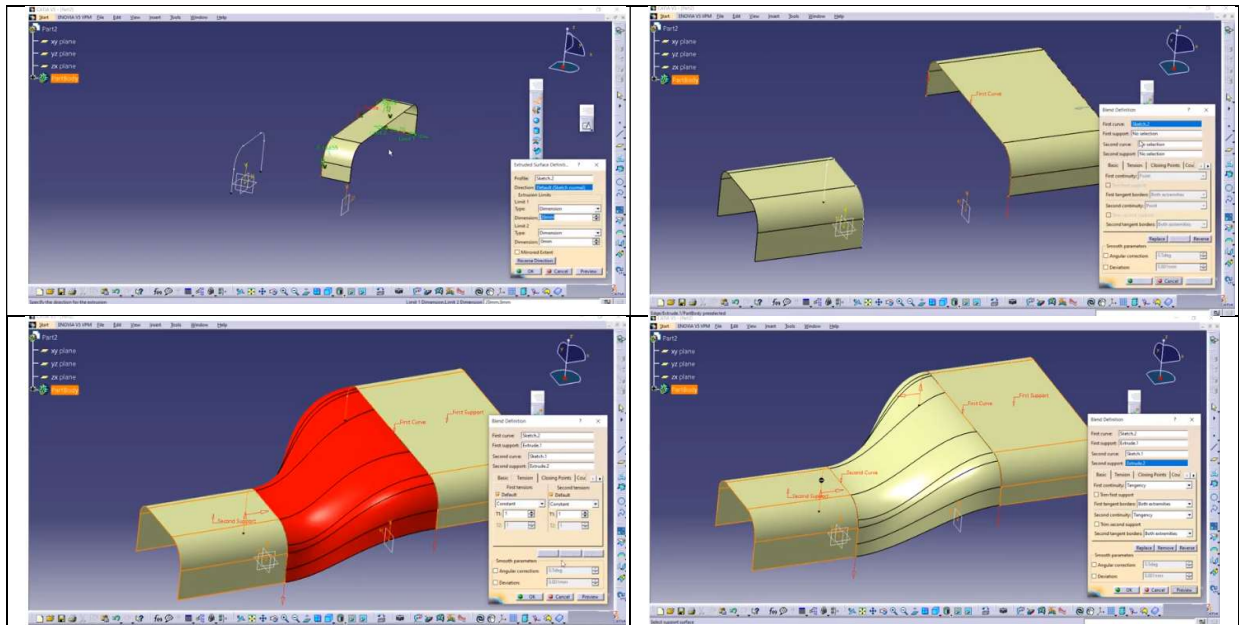
Reference

<https://www.youtube.com/watch?v=q3AGW6VCI18>

Experiment No: 6

Date:

Aim: To create wireframe, surface and solid models



wireframe, surface and solid models

Procedure:

1. Open file and save as Blend, Select Y-Z plane
2. Select Part body, Sketch the rectangular with corner curves
3. Offset the blend and zoom and create one more rectangle same with bigger dimensions
4. Extruded surface design with Two surfaces of two rectangles
5. Select both rectangle and blend both surfaces.

Reference

<https://www.youtube.com/watch?v=e9I1hyktTHQ>

INTRODUCTION TO CAM

Computer-aided manufacturing (CAM) is the use of computer-based software tools that assist engineers and machinists in manufacturing or prototyping product components. Its primary purpose is to create a faster production process and components with more precise dimensions and material consistency, which in some cases, uses only the required amount of raw material (thus minimizing waste), while simultaneously reducing energy consumption. CAM is a programming tool that makes it possible to manufacture physical models using computer-aided design (CAD) programs. CAM creates real life versions of components designed within a software package

CNC Technology:

Numerical Control (NC) is a software-based machine tool control technique developed at Massachusetts Institute of Technology (MIT) in early 1960s. It has now evolved into a mature technology known as Computer Numerical Control (CNC). Although major applications of CNC even today continue to be in machining, it finds applications in other processes such as sheet metal working, non-traditional machining and inspection. Robots and Rapid Prototyping machines are also CNC controlled. In fact, any process that can be visualized as a sequence of motions and switching functions can be controlled by CNC. These motions and switching functions are input in the form of alphanumeric instructions. CNC is the basis of flexible automation which helps industries cut down time-to-market and enables launch of even low volume products. Unlimited muscle power, unmanned operation, independent axes coordinated through software, simplified generic tooling even for the most complex jobs and accurate construction are some of the salient features of CNC.

CNC Machining:

Automats and Special Purpose Machines (SPMs) require special cams/ templates and clutch settings for each part. Manufacture of these cams/ templates is costly and slow. Furthermore, changing over from one part to the other on these machines also consumes considerable time. The high cost and long time of these hard automated machines to produce parts can be justified only in mass production. With the advent of fast, rigid and accurate CNC machines and sophisticated CAM packages, generation of NC programs and change over from one product to the other are easy and fast as it does not require any mechanical change. These in conjunction with advanced cutting tools have made High Speed Cutting (HSC) of hard materials a reality. Therefore, CNC machining has become a standard means to produce dies and molds; tool makers today require EDM only for producing inaccessible regions, sharp corners, tiny features and desired surface quality. Intricate aerospace parts are realized through 5 axis CNC machining. Internet technology in a global village enables designing in one place, NC programming and verification in another place and actual machining in yet another place.

Advantages of CNC

- Flexibility
- Accuracy
- Speed
- Simplified fixturing and generic cutting tools
- Storage of machining skill in NC programs
- Less skilled operators will do
- Less fatigue to the operators

G-codes:

G-Code, or preparatory code or function, are functions in the Numerical control programming language. The G-codes are the codes that position the tool and do the actual work, as opposed to M-codes, that manages the machine; T for tool-related codes. S and F are tool-Speed and tool-Feed, and finally D-codes for tool compensation. The programming language of Numerical Control (NC) is sometimes informally called G-code. But in actuality, G-codes are only a part of the NC-programming language that controls NC and CNC machine tools. A basic list of 'G' operation codes is given below. This direct motion of the tool.

1. G00 - Rapid move (not cutting)
2. G01 - Linear move
3. G02 - Clockwise circular motion
4. G03 - Counterclockwise circular motion
5. G04 - Dwell
6. G05 - Pause (for operator intervention)
7. G08 - Acceleration
8. G09 - Deceleration
9. G17 - x-y plane for circular interpolation
10. G18 - z-x plane for circular interpolation
11. G19 - y-z plane for circular interpolation
12. G20 - turning cycle or inch data specification
13. G21 - thread cutting cycle or metric data specification
14. G24 - face turning cycle
15. G25 - wait for input #1 to go low (Prolight Mill)
16. G26 - wait for input #1 to go high (Prolight Mill)
17. G28 - return to reference point
18. G29 - return from reference point
19. G31 - Stop on input (INROB1 is high) (Prolight Mill)
20. G33-35 - thread cutting functions (Emco Lathe)
21. G35 - wait for input #2 to go low (Prolight Mill)
22. G36 - wait for input #2 to go high (Prolight Mill)
23. G40 - cutter compensation cancel
24. G41 - cutter compensation to the left
25. G42 - cutter compensation to the right
26. G43 - tool length compensation, positive
27. G44 - tool length compensation, negative
28. G50 - Preset position
29. G70 - set inch based units or finishing cycle
30. G71 - set metric units or stock removal

31. G72 - indicate finishing cycle (EMCO Lathe)
32. G72 - 3D circular interpolation clockwise (Prolight Mill)
33. G73 - turning cycle contour (EMCO Lathe)
34. G73 - 3D circular interpolation counter clockwise (Prolight Mill)
35. G74 - facing cycle contour (Emco Lathe)
36. G74.1 - disable 360 deg arcs (Prolight Mill)
37. G75 - pattern repeating (Emco Lathe)
38. G75.1 - enable 360 degree arcs (Prolight Mill)
39. G76 - deep hole drilling, cut cycle in z-axis
40. G77 - cut-in cycle in x-axis
41. G78 - multiple threading cycle
42. G80 - fixed cycle cancel
43. G81-89 - fixed cycles specified by machine tool manufacturers
44. G81 - drilling cycle (Prolight Mill)
45. G82 - straight drilling cycle with dwell (Prolight Mill)
46. G83 - drilling cycle (EMCO Lathe)
47. G83 - peck drilling cycle (Prolight Mill)
48. G84 - taping cycle (EMCO Lathe)
49. G85 - reaming cycle (EMCO Lathe)
50. G85 - boring cycle (Prolight mill)
51. G86 - boring with spindle off and dwell cycle (Prolight Mill)
52. G89 - boring cycle with dwell (Prolight Mill)
53. G90 - absolute dimension program
54. G91 - incremental dimensions
55. G92 - Spindle speed limit
56. G93 - Coordinate system setting
57. G94 - Feed rate in ipm (EMCO Lathe)
58. G95 - Feed rate in ipr (EMCO Lathe)
59. G96 - Surface cutting speed (EMCO Lathe)
60. G97 - Rotational speed rpm (EMCO Lathe)
61. G98 - withdraw the tool to the starting point or feed per minute
62. G99 - withdraw the tool to a safe plane or feed per revolution
63. G101 - Spline interpolation (Prolight Mill)

M-Codes:

M-Codes control machine functions and these include,

1. M00 - program stop
2. M01 - optional stop using stop button
3. M02 - end of program
4. M03 - spindle on CW
5. M04 - spindle on CCW
6. M05 - spindle off
7. M06 - tool change
8. M07 - flood with coolant
9. M08 - mist with coolant
10. M08 - turn on accessory #1 (120VAC outlet) (Prolight Mill)
11. M09 - coolant off
12. M09 - turn off accessory #1 (120VAC outlet) (Prolight Mill)
13. M10 - turn on accessory #2 (120VAC outlet) (Prolight Mill)
14. M11 - turn off accessory #2 (120VAC outlet) (Prolight Mill) or tool change

15. M17 - subroutine end
16. M20 - tailstock back (EMCO Lathe)
17. M20 - Chain to next program (Prolight Mill)
18. M21 - tailstock forward (EMCO Lathe)
19. M22 - Write current position to data file (Prolight Mill)
20. M25 - open chuck (EMCO Lathe)
21. M25 - set output #1 off (Prolight Mill)
22. M26 - close chuck (EMCO Lathe)
23. M26 - set output #1 on (Prolight Mill)
24. M30 - end of tape (rewind)
25. M35 - set output #2 off (Prolight Mill)
26. M36 - set output #2 on (Prolight Mill)
27. M38 - put stepper motors on low power standby (Prolight Mill)
28. M47 - restart a program continuously, or a fixed number of times (Prolight Mill)
29. M71 - puff blowing on (EMCO Lathe)
30. M72 - puff blowing off (EMCO Lathe)
31. M96 - compensate for rounded external curves
32. M97 - compensate for sharp external curves
33. M98 - subprogram call
34. M99 - return from subprogram, jump instruction
35. M101 - move x-axis home (Prolight Mill)
36. M102 - move y-axis home (Prolight Mill)
37. M103 - move z-axis home (Prolight Mill)

CNC PROGRAMMING:

The coordinates are almost exclusively Cartesian and the origin is on the work piece. For a lathe, the infeed/radial axis is the x-axis, the carriage/length axis is the z-axis. There is no need for a y-axis because the tool moves in a plane through the rotational center of the work. Coordinates on the work piece shown below are relative to the work.

CNC lathes are rapidly replacing the older production lathes (multispindle, etc) due to their ease of setting and operation. They are designed to use modern carbide tooling and fully utilize modern processes. The part may be designed and the tool paths programmed by the CAD/CAM process, and the resulting file uploaded to the machine, and once set and trailed the machine will continue to turn out parts under the occasional supervision of an operator. The machine is controlled electronically via a computer menu style interface; the program may be modified and displayed at the machine, along with a simulated view of the process. The setter/operator needs a high level of skill to perform the process, however the knowledge base is broader compared to the older production machines where intimate knowledge of each machine was considered essential. These machines are often set and operated by the same person, where the operator will supervise a small number of machines (cell).

EXP NO: 7

DATE:

STUDY OF "G" CODES AND "M" CODES

Aim: To study the "G" codes and "M" codes for the CNC lathe and milling machine.

Programming codes:

- Codes are modal and do not have to be repeated in every sequence line.
- All dimensions are entered as decimals.

G –codes for turning machine:

MTAB INDIA PVT LTD

G- To define tool movement for preparatory function.

- G00 - Rapid move (not cutting)
- G01 - Linear move
- G02 - Clockwise circular motion
- G03 - Counterclockwise circular motion
- G04 - Dwell
- G17 - x-y plane
- G18 - z-x plane
- G19 - y-z plane
- G20 - inch data specification
- G21 - metric data specification
- G28 - return to reference point (home position)
- G40 - cutter compensation cancel
- G50 - Preset position
- G70 - set inch based units or finishing cycle
- G71 - set metric units or stock removal or multiple turning
- G75 - pattern repeating (Emco Lathe) or multiple grooving
- G76 - deep hole drilling, cut cycle in z-axis or multiple threading
- G90 - absolute dimension program
- G91 - incremental dimensions
- G98 - withdraw the tool to the starting point or feed per minute

M –codes for turning machine:

- M00 - program stop
- M01 - optional stop using stop button
- M02 - end of program
- M03 - spindle on CW
- M04 - spindle on CCW
- M05 - spindle off
- M06 - tool change
- M10 - chuck open
- M11 - chuck close or tool change
- M30 - program stop and restart

G –codes for milling machine:

MTAB INDIA PVT LTD

- G00 - Rapid move (not cutting)
- G01 - Linear move
- G02 - Clockwise circular motion
- G03 - Counterclockwise circular motion
- G04 - Dwell
- G17 - x-y plane
- G18 - z-x plane
- G19 - y-z plane
- G20 - inch data specification
- G21 - metric data specification
- G28 - return to reference point (home position)
- G40 - cutter compensation cancel
- G50 - Preset position
- G68 - Rotation ON
- G69 - Rotation OFF
- G80 - fixed cycle cancel
- G83 - drilling or peak drilling cycle
- G90 - absolute dimension program
- G91 - incremental dimensions
- G94 - Feed rate in ipm
- G98 - withdraw the tool to the starting point or feed per minute
- G99 - withdraw the tool to a safe plane or feed per revolution
- G172- pocket frame mill
- G173- pocket outside face mill

M -codes for milling machine:

- M00 - program stop
- M01 - optional stop using stop button
- M02 - end of program
- M03 - spindle on CW
- M04 - spindle on CCW
- M05 - spindle off
- M06 - tool change
- M10 - chuck open
- M11 - chuck close or tool change
- M30 - program stop and restart
- M98 - sub program call
- M99 - sub program exit

Result:

EXP NO: 8

DATE:

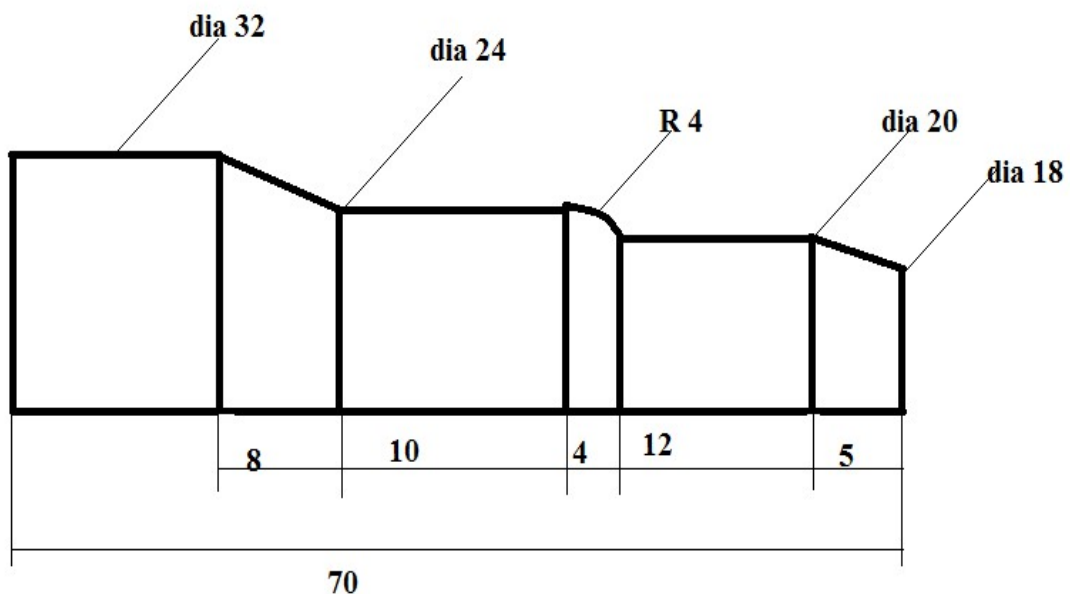
CNC LATHE – MULTIPLE TURNING OPERATION

Aim: To perform multiple turning operation by using CNC programming codes.

Apparatus required:

- CNC machine
- Personal computer
- CNC software
- Cutting tool
- Work piece

Program:



```
Define the work piece  
Work piece("cylinder");  
G71G94;  
G75X0Z0;  
M06T2D1;  
M03S1500;  
M08;  
G00X32Z2;
```

Go for counter turning-counter call-sub program-give the name-then back-stock removal-there is a window for stock removal appears-

| | |
|-------------|------------------------------------|
| PRG | B |
| SC | 2.00 |
| F | 100.00 |
| MACHINING | ROUGHING+FINISHING |
| FS | 100.00 LONGITUDINAL OUT SIDE |
| D | 0.300 |
| UX | 0.100 |
| UZ | 0.100 |
| XD | 32.00 abs |
| ZD | 2.00 abs |
| RELIEF CUTS | YES |
| FR | 100.00 |

ACCEPT

Note: the stock removal name is different what we given in the sub program name.

```
M09;
G75X0Z0;
M05;
M30;
```

Then go subprogram in the part program it self-subprogram-new- the name is same as what we given in the countercall sub program name- then type the program.

```
G01X18 Z0;
G01 X20 Z-5;
G01Z-17;
G03X24Z-21CR=4;
G01Z-31;
G01X32 Z-39;
M17;
```

Then go for the main program-simulation-execute-auto-reset-start.

Result:

EXP NO: 9

DATE:

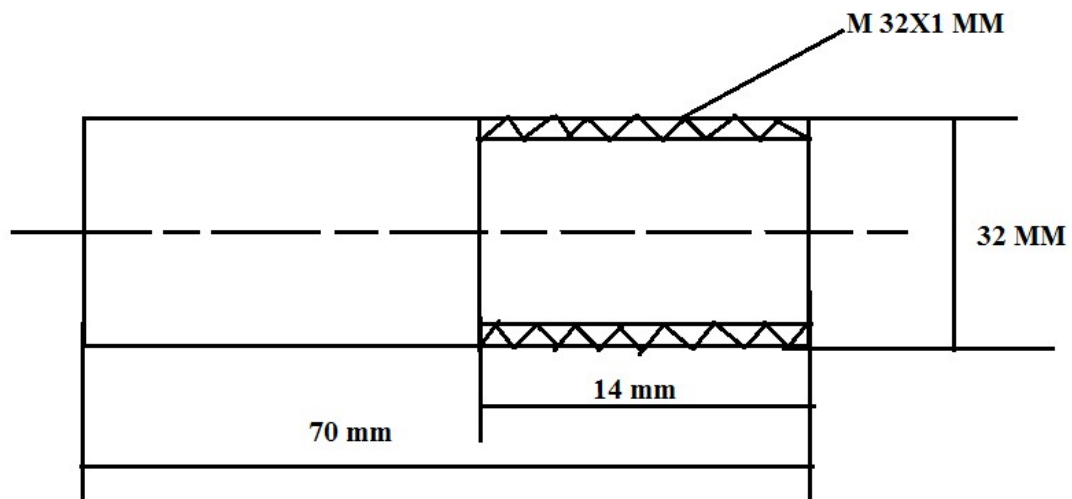
CNC LATHE – EXTERNAL THREADING OPERATION

Aim: To perform external threading operation by using CNC programming codes.

Apparatus required:

- CNC machine
- Personal computer
- CNC software
- Cutting tool
- Work piece

Program:



Define the work piece
Work piece("cylinder");
G71G94;
G75X0Z0;
M06T04D1;
M03S500;
M08;
G00X32Z2;

Then go for turning in the panel board-and then threading table will be appears.

| | |
|-----------|---|
| TABLE | ISO METRIC |
| SELECT | M 10 |
| P | 1.500 mm/rev |
| MACHINING | ROUGHING+FINISHING Linear External thread |

| | |
|----------|------------|
| X0 | 32.00 |
| Z0 | 0.000 |
| Z1 | -14.00 abs |
| Lw | 5.000 |
| LR | 0.000 |
| H1 | 0.920 |
| ∞p | 30.00 |
| D1 | 0.100 |
| U | 0.010 |
| NN | 0 |
| VR | 0.500 |
| MULTIPLE | NO |
| ∞0 | 0.00 |

ACCEPT

M09;
G75X0Z0;
M05;
M30;

Then go for the main program-simulation-execute-auto-reset-start.

Result:

EXP NO: 10

DATE:

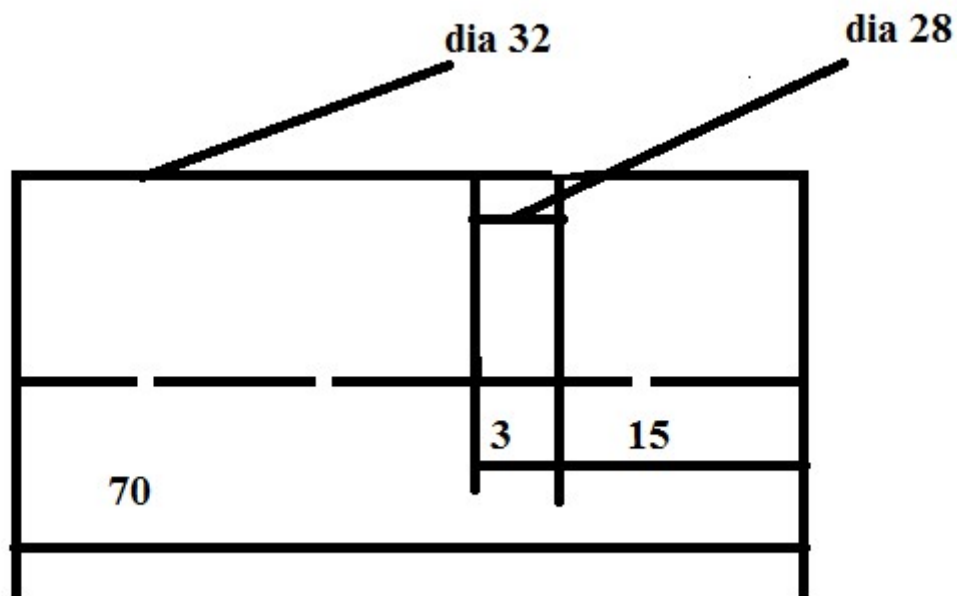
CNC LATHE – EXTERNAL GROOVING OPERATION

Aim: To perform external Grooving operation by using CNC programming codes.

Apparatus required:

- CNC machine
- Personal computer
- CNC software
- Cutting tool
- Work piece

Program:



```
Define the work piece  
Work piece("cylinder");  
G71G94;  
G75X0Z0;  
M06T6D1;  
M03S900;  
M08;  
G00X33Z5;  
G01 X32 Z-15F50;
```

Then go for turning in the panel board-Grooving-and then grooving table will be appears.

| | |
|-----------|--|
| SC | 5.00 |
| F | 50.00 |
| MACHINING | ROUGHING+FINISHING |
| POS | Position of machining reference point |
| X0 | 32.00 |
| Z0 | -17.5 |
| B1 | 3.00 |
| T1 | 28.00 |
| D | 0.100 |
| UX | 0.050 |
| UZ | 0.050 |
| N | 1 |

ACCEPT

G01 Z5;
M09;
G75X0Z0;
M05;
M30;

Then go for the main program-simulation-execute-auto-reset-start.

Result:

EXP NO: 11

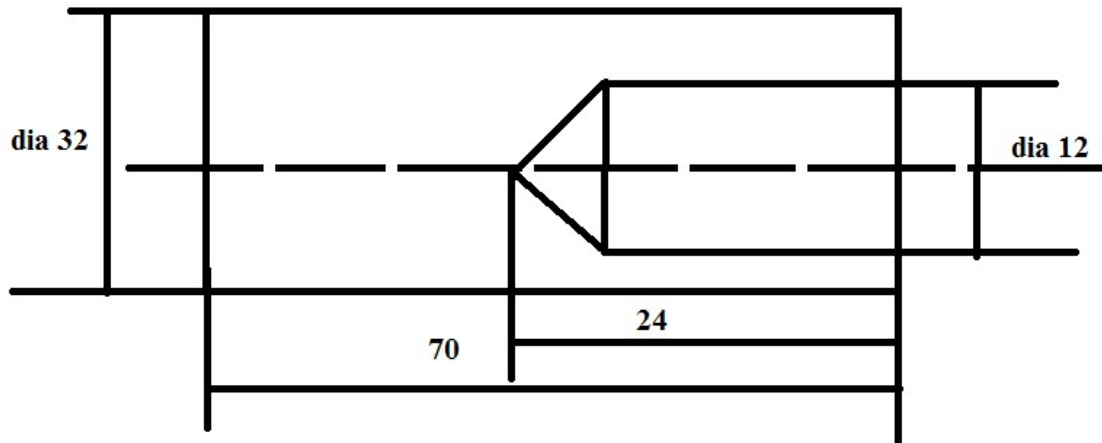
DATE:
CNC LATHE – DRILLING OPERATION

Aim: To perform Drilling operation by using CNC programming codes.

Apparatus required:

- CNC machine
- Personal computer
- CNC software
- Cutting tool
- Work piece

Program:



Define the work piece
Work piece("cylinder");
G71G94;
G75X0Z0;
M06T7D1;
M03S1000;
M08;
G00X0Z5F50;

Then go for DRILLING in the panel board-Deep hole drilling-and then Drilling table will be appears.

| | |
|---------------|---------------|
| PL | G17(XY) |
| RP | 1.00 |
| SC | 5.00 |
| Z0 | 0.00 |
| Z1 | -24 abs |
| D | -0.500 abs |
| FD1 | 70.00% |
| DF | 50.00% |
| V1 | 0.300 |
| Lead distance | Automatically |
| DTB | 0.000S |
| DT | 0.000S |
| DTB | 0.000S |

**ACCEPT
THEN TYPE
MCALL:**

G00 Z10;
M09;
G75X0Z0;
M05;
M30;

Then go for the main program-simulation-execute-auto-reset-start.

Result:

EXP NO: 12

DATE:

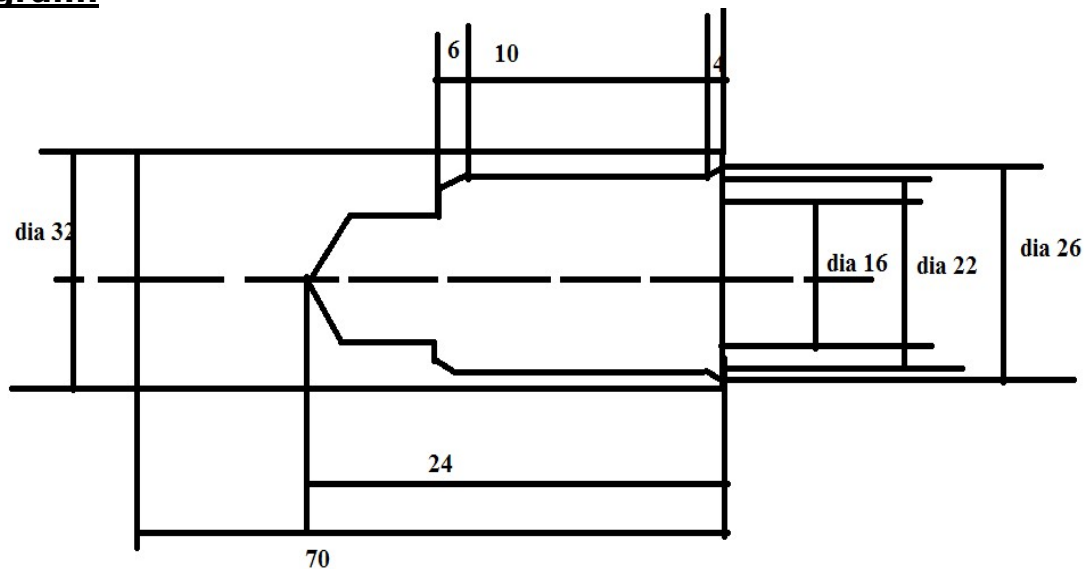
CNC LATHE – BORING OR INTERNAL TURNING OPERATION

Aim: To perform internal turning operation by using CNC programming codes.

Apparatus required:

- CNC machine
- Personal computer
- CNC software
- Cutting tool
- Work piece

Program:



Define the work piece
Work piece("cylinder");
G71G94;
G75X0Z0;
M06T3D1;
M03S1500;
M08;
G00X12Z5F100;

Go for counter turning-counter call-sub program-give the name-then back-stock removal-there is a window for stock removal appears-

| | |
|-------------|----------------------------------|
| PRG | G |
| SC | 5.00 |
| F | 100.00 |
| MACHINING | ROUGHING+FINISHING |
| FS | 100.00 LONGITUDINAL INSIDE |
| RP | 0.500abs |
| D | 0.200 |
| UX | 0.100 |
| UZ | 0.100 |
| XD | 12.00 abs |
| ZD | 5.00 abs |
| RELIEF CUTS | YES |
| FR | 100.00 |

ACCEPT

Note: the stock removal name is different what we given in the sub program name.

M09;

G75X0Z0;

M05;

M30;

Then go subprogram in the part program it self-subprogram-new-the name is same as what we given in the countercall sub program name-then type the program.

G01X26 Z0;

G02 X22 Z-4CR=4;

G01Z-14;

G01X16Z-20;

G01X12;

M17;

Then go for the main program-simulation-execute-auto-reset-start.

Result:

EXP NO: 13

DATE:

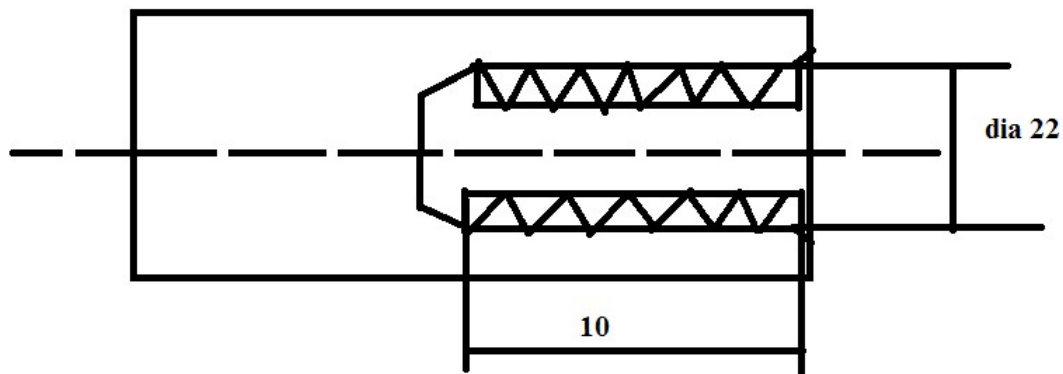
CNC LATHE – INTERNAL THREADING OPERATION

Aim: To perform Internal threading operation by using CNC programming codes.

Apparatus required:

- CNC machine
- Personal computer
- CNC software
- Cutting tool
- Work piece

Program:



Define the work piece
Work piece("cylinder");
G71G94;
G75X0Z0;
M06T01D1;
M03S500;
M08;
G00X22Z2;

Then go for turning in the panel board-and then threading table will be appears.

| | |
|-----------|---|
| TABLE | ISO METRIC |
| SELECT | M 10 |
| P | 1.500 mm/rev |
| MACHINING | ROUGHING+FINISHING Linear Internal thread |
| X0 | 22.00 |

| | |
|----------|---------|
| Z0 | 0.000 |
| Z1 | -10 abs |
| Lw | 5.000 |
| LR | 0.000 |
| H1 | 0.920 |
| ∞p | 30.00 |
| D1 | 0.100 |
| U | 0.010 |
| NN | 0 |
| VR | 0.500 |
| MULTIPLE | NO |
| ∞0 | 0.00 |

ACCEPT

M09;
G75X0Z0;
M05;
M30;

Then go for the main program-simulation-execute-auto-reset-start.

Result:

Experiment 11 Program for Pick and Place activity:

| STATEMENT | STATEMENT DESCRIPTION |
|--------------|--|
| BRANCH PICK | The branch of program indicating part picks. |
| MOVE INTER | Move to an intermediate position chute. |
| WAIT 12 | Wait for an incoming part to chute. |
| SIGNAL 5 | Open gripper fingers (Sensor control) |
| MOVE PICK-UP | Move gripper and Pick-up the object. |
| SIGNAL 6 | Close the gripper to grasp the object |
| MOVE INTER | Depart to intermediate position above chute. |
| END BRANCH | End of pick-up activity. |
| BRANCH PLACE | Start of placing activity |
| MOVE Z (-50) | Position part and gripper above the pallet |
| SIGNAL 5 | Open gripper to release the part |
| MOVE Z (50) | Depart from the place point. |
| END BRANCH | |

CONCLUSION: Thus, we have studied how to perform Pick and Place operation.