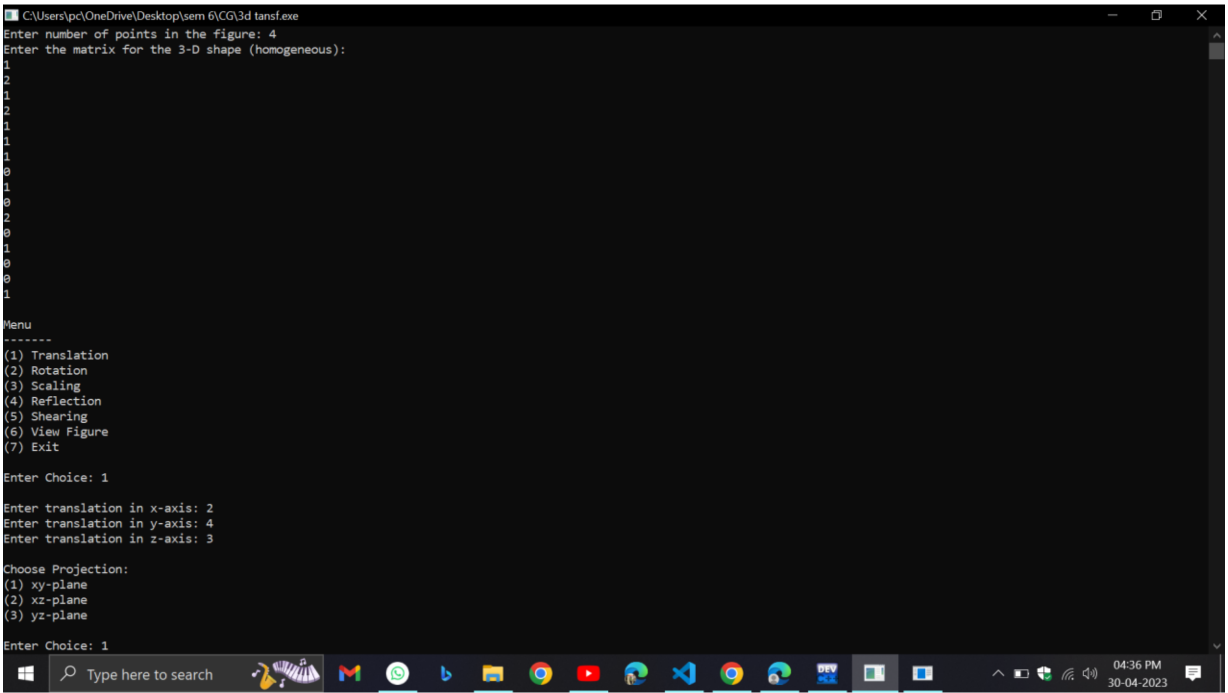
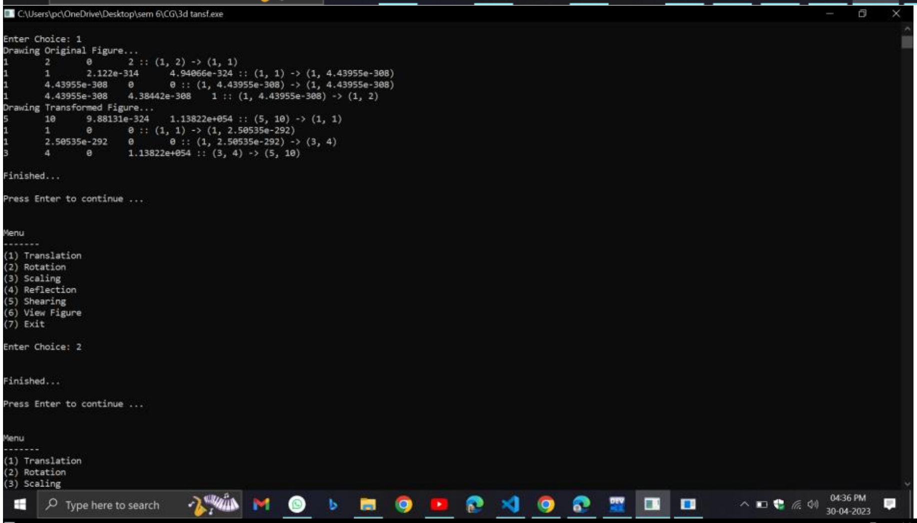
```
#include<iostream>
#define _USE_MATH_DEFINES
#include <conio.h>
#include<graphics.h>
#include<math.h>
#include <stdio.h>
#include <stdlib.h>
#define COORD_SHIFT 100
using namespace std;
double **inputFigure(int n)
{
  cout << "Enter the matrix for the 3-D shape (homogeneous):\n";

  double **figure = NULL;
  figure = new double *[n];

  for (int i = 0; i < n; i++)
  {
    figure[i] = new double[4];
    for (int j = 0; j < 4; j++)
    {
      cin >> figure[i][j];
    }
  }

  return figure;
}

void drawFigure(double **points, int n, int p)
{
  int a, b;
  switch (p)
  {
  case 1:
    a = 0;
    b = 1;
    break;
  case 2:
    a = 0;
    b = 2;
    break;
  case 3:
    a = 1;
    b = 2;
    break;
  }

  setcolor(WHITE);

  for (int i = 0; i < n; i++)
  {
    line(COORD_SHIFT + points[i][a],
         COORD_SHIFT + points[i][b],
         COORD_SHIFT + points[(i + 1) % n][a],
         COORD_SHIFT + points[(i + 1) % n][b]);

    cout << points[i][0] << "\t"
         << points[i][1] << "\t"
         << points[i][2] << "\t"
         << points[i][3] << " "
         << ":: (" << points[i][a] << ", " << points[i][b] << ") "
         << "-> (" << points[(i + 1) % n][a] << ", " << points[(i + 1) % n][b] << ")"
         << endl;
  }

  delay(5e3);
  cleardevice();
}

double **translate(double **figure, int dim, int l, int m, int n)
{
  double **_figure = NULL;
  int T[dim][4] = {{1, 0, 0, 0},
                   {0, 1, 0, 0},
                   {0, 0, 1, 0},
                   {l, m, n, 1}};

  _figure = new double *[dim];

  for (int i = 0; i < dim; i++)
  {
    _figure[i] = new double[4];
    for (int j = 0; j < 4; j++)
    {
      for (int k = 0; k < dim; k++)
      {
        _figure[i][j] += figure[i][k] * T[k][j];
      }
    }
  }

  return _figure;
}

double **rotate(double **figure, int dim, double theta)
{
  double **_figure = NULL;
  double T[dim][3] = {{cos(theta * M_PI / 180.0), sin(theta * M_PI / 180.0), 0},
                      {-sin(theta * M_PI / 180.0), cos(theta * M_PI / 180.0), 0},
                      {0, 0, 1}};

  _figure = new double *[dim];
  for (int i = 0; i < dim; i++)
  {
    _figure[i] = new double[3];
    for (int j = 0; j < 2; j++)
    {
      for (int k = 0; k < dim; k++)
      {
        _figure[i][j] += figure[i][k] * T[k][j];
      }
    }
  }

  return _figure;
}

double **scale(double **figure, int dim, double l, double m, double n)
{
  double **_figure = NULL;
  double T[dim][4] = {{l, 0, 0, 0},
                      {0, m, 0, 0},
                      {0, 0, n, 0},
                      {0, 0, 0, 1}};

  _figure = new double *[dim];

  for (int i = 0; i < dim; i++)
  {
    _figure[i] = new double[4];
    for (int j = 0; j < 4; j++)
    {
      for (int k = 0; k < dim; k++)
      {
        _figure[i][j] += figure[i][k] * T[k][j];
      }
    }
  }

  return _figure;
}

double **scale(double **figure, int dim, double s)
{
  double **_figure = NULL;
  double T[dim][4] = {{1, 0, 0, 0},
                      {0, 1, 0, 0},
                      {0, 0, 1, 0},
                      {0, 0, 0, s}};

  _figure = new double *[dim];
  for (int i = 0; i < dim; i++)
  {
    _figure[i] = new double[4];
    for (int j = 0; j < 4; j++)
    {
      for (int k = 0; k < dim; k++)
      {
        _figure[i][j] += figure[i][k] * T[k][j];
      }
    }
  }

  return _figure;
}

double **reflect(double **figure, int dim, int c)
{
  double **_figure = NULL;
  int T[dim][3] = {{1, 0, 0}, {0, 1, 0}, {0, 0, 1}};

  switch (c)
  {
  case 1:
    T[1][1] = -1;
    break;
  case 2:
    T[0][0] = -1;
    break;
  case 3:
    T[0][0] = 0;
    T[0][1] = 1;
    T[1][0] = 1;
    T[1][1] = 0;
    break;
  case 4:
    T[0][0] = -1;
    T[1][1] = -1;
    break;
  default:
    return NULL;
    break;
  }

  _figure = new double *[dim];

  for (int i = 0; i < dim; i++)
  {
    _figure[i] = new double[3];
    for (int j = 0; j < 3; j++)
    {
      for (int k = 0; k < dim; k++)
      {
        _figure[i][j] += figure[i][k] * T[k][j];
      }
    }
  }

  return _figure;
}

double **shear(double **figure, int dim, int m, int n)
{
  double **_figure = NULL;
  int T[dim][3] = {{1, n, 0}, {m, 1, 0}, {0, 0, 1}};

  _figure = new double *[dim];

  for (int i = 0; i < dim; i++)
  {
    _figure[i] = new double[3];
    for (int j = 0; j < 3; j++)
    {
      for (int k = 0; k < dim; k++)
      {
        _figure[i][j] += figure[i][k] * T[k][j];
      }
    }
  }

  return _figure;
}

double **project(double **figure, int dim, int p)
{
  double **_figure = NULL;
  int P[dim][4] = {{1, 0, 0, 0},
                  {0, 1, 0, 0},
                  {0, 0, 1, 0},
                  {0, 0, 0, 1}};

  switch (p)
  {
  case 1:
    P[2][2] = 0;
    break;
  case 2:
    P[1][1] = 0;
    break;
  case 3:
    P[0][0] = 0;
    break;
  }
  _figure = new double *[dim];

  for (int i = 0; i < dim; i++)
  {
    _figure[i] = new double[4];
    for (int j = 0; j < 4; j++)
    {
      for (int k = 0; k < dim; k++)
      {
        _figure[i][j] += figure[i][k] * P[k][j];
      }
    }
  }

  return _figure;
}

void menu(double **figure, int dim)
{
  int ch = 0;
  double l, m, n, p;
  double **_figure, **_projected;

  do
  {
    //clrscr();
    cout << "\nMenu\n-------\n(1) Translation\n(2) Rotation";
    cout << "\n(3) Scaling\n(4) Reflection\n(5) Shearing";
    cout << "\n(6) View Figure\n(7) Exit\n\nEnter Choice: ";
    cin >> ch;
    cout << endl;
    switch (ch)
    {
    case 1:

      cout << "Enter translation in x-axis: ";
      cin >> l;
      cout << "Enter translation in y-axis: ";
      cin >> m;
      cout << "Enter translation in z-axis: ";
      cin >> n;

      _figure = translate(figure, dim, l, m, n);

      cout << "\nChoose Projection:\n(1) xy-plane\n(2) xz-plane\n(3) yz-plane\n"
           << "\nEnter Choice: ";
      cin >> p;

      if (p > 3 || p < 1)
      {
```

```
    cout << "\nInvalid Projection!";
    cin.ignore();
    cin.get();
    continue;
  }

  cout << "Drawing Original Figure...\n";
  drawFigure(project(figure, dim, p), dim, p);

  cout << "Drawing Transformed Figure...\n";
  drawFigure(project(_figure, dim, p), dim, p);
  break;

case 3:
  int scalingCh;
  cout << "Scaling:\n(1) Overall Scaling\n(2) Local Scaling\n\nEnter Choice: ";
  cin >> scalingCh;

  switch (scalingCh)
  {
  case 1:
    cout << "Enter scaling factor: ";
    cin >> l;
    _figure = scale(figure, dim, l);
    break;

  case 2:
    cout << "Enter scaling in x-axis: ";
    cin >> l;
    cout << "Enter scaling in y-axis: ";
    cin >> m;
    cout << "Enter scaling in z-axis: ";
    cin >> n;
    _figure = scale(figure, dim, l, m, n);
    break;
  }

  cout << "Drawing Original Figure...\n";
  drawFigure(project(figure, dim, p), dim, p);

  cout << "Drawing Transformed Figure...\n";
  drawFigure(project(_figure, dim, p), dim, p);
  break;



case 6:

  cout << "\nChoose Projection:\n(1) xy-plane\n(2) xz-plane\n(3) yz-plane\n"
       << "\nEnter Choice: ";
  cin >> p;
```

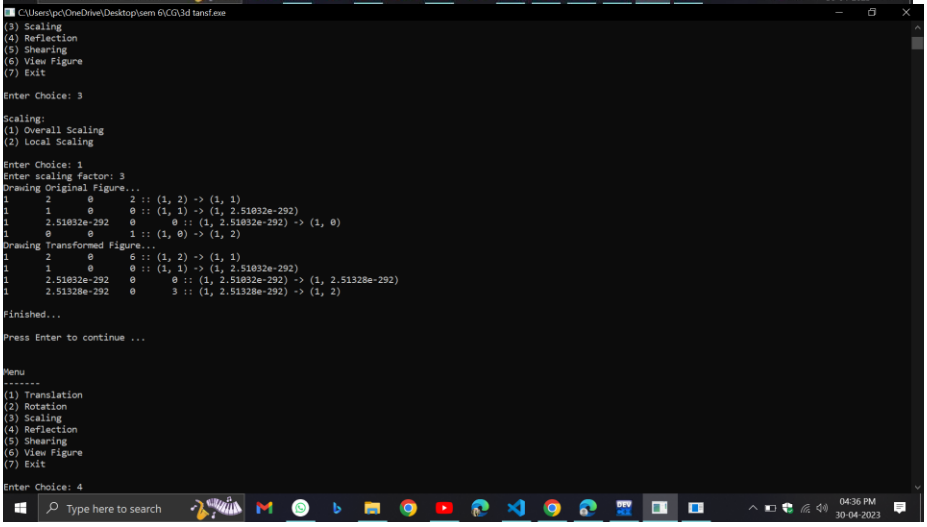
```
      if (p > 3 || p < 1)
      {
        cout << "\nInvalid Projection!";
        cin.ignore();
        cin.get();
        continue;
      }

      cout << "Drawing Original Figure...\n";
      drawFigure(project(figure, dim, p), dim, p);
    case 7:
    default:
      break;
    }

    if (ch != 6)
      delete _figure;

    cout << endl
         << "Finished..."
         << endl;

    if (ch != 7)
    {
      cout << "\nPress Enter to continue ...\n";
      cin.ignore();
      cin.get();
    }
  } while (ch != 7);
};

int main(void)
{
  int n;
  double **fig;
  int gd = DETECT, gm;

  initgraph(&gd, &gm, NULL);

  cout << "Enter number of points in the figure: ";
  cin >> n;

  fig = inputFigure(n);

  menu(fig, n);

  delete fig;
  closegraph();
```



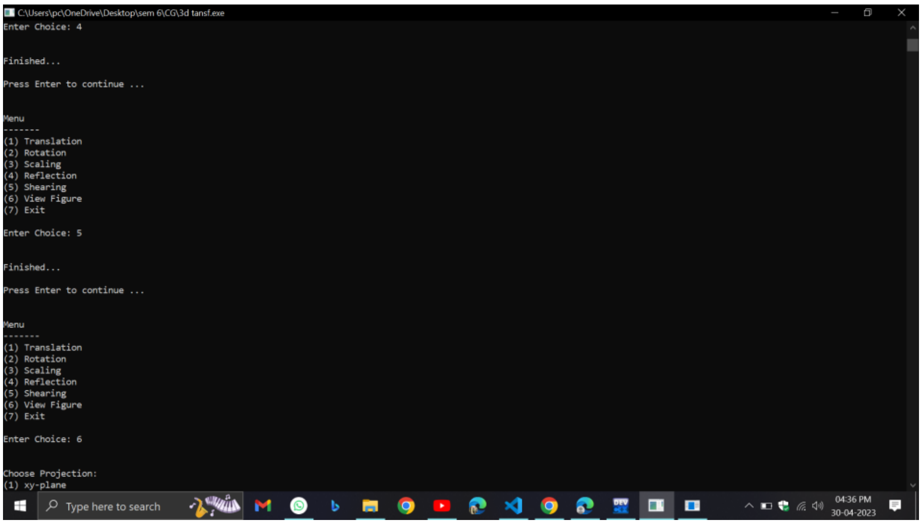
```
Enter number of points in the figure: 4
Enter the matrix for the 3-D shape (homogeneous):
1
2
1
2
1
1
1
0
1
0
2
0
1
0
0
1

Menu
-------
(1) Translation
(2) Rotation
(3) Scaling
(4) Reflection
(5) Shearing
(6) View Figure
(7) Exit

Enter Choice: 1

Enter translation in x-axis: 2
Enter translation in y-axis: 4
Enter translation in z-axis: 3

Choose Projection:
(1) xy-plane
(2) xz-plane
(3) yz-plane

Enter Choice: 1
```



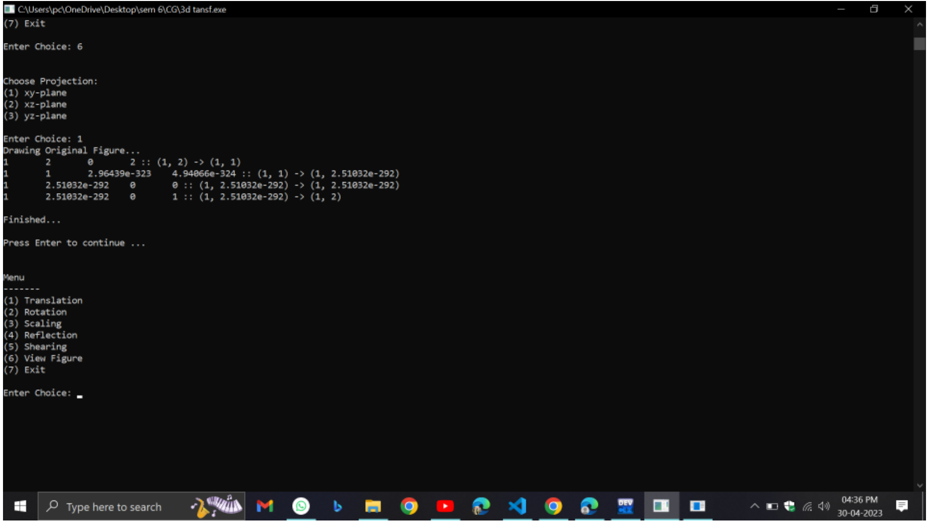
```
Enter Choice: 1
Drawing Original Figure...
1       2       0       2 :: (1, 2) -> (1, 1)
1       1       2.122e-314      4.94066e-324 :: (1, 1) -> (1, 4.43955e-308)
1       4.43955e-308    0       0 :: (1, 4.43955e-308) -> (1, 4.43955e-308)
1       4.43955e-308    4.38442e-308    1 :: (1, 4.43955e-308) -> (1, 2)
Drawing Transformed Figure...
5       10      9.88131e-324    1.13822e+054 :: (5, 10) -> (1, 1)
1       1       0       0 :: (1, 1) -> (1, 2.50535e-292)
1       2.50535e-292    0       0 :: (1, 2.50535e-292) -> (3, 4)
3       4       0       1.13822e+054 :: (3, 4) -> (5, 10)

Finished...

Press Enter to continue ...


Menu
-------
(1) Translation
(2) Rotation
(3) Scaling
(4) Reflection
(5) Shearing
(6) View Figure
(7) Exit

Enter Choice: 2

Finished...

Press Enter to continue ...


Menu
-------
(1) Translation
(2) Rotation
(3) Scaling
```



```
(3) Scaling
(4) Reflection
(5) Shearing
(6) View Figure
(7) Exit

Enter Choice: 3

Scaling:
(1) Overall Scaling
(2) Local Scaling

Enter Choice: 1
Enter scaling factor: 3
Drawing Original Figure...
1       2       0       2 :: (1, 2) -> (1, 1)
1       1       0       0 :: (1, 1) -> (1, 2.51032e-292)
1       2.51032e-292    0       0 :: (1, 2.51032e-292) -> (1, 0)
1       0       0       1 :: (1, 0) -> (1, 2)
Drawing Transformed Figure...
1       2       0       6 :: (1, 2) -> (1, 1)
1       1       0       0 :: (1, 1) -> (1, 2.51032e-292)
1       2.51032e-292    0       0 :: (1, 2.51032e-292) -> (1, 2.51328e-292)
1       2.51328e-292    0       3 :: (1, 2.51328e-292) -> (1, 2)

Finished...

Press Enter to continue ...


Menu
-------
(1) Translation
(2) Rotation
(3) Scaling
(4) Reflection
(5) Shearing
(6) View Figure
(7) Exit

Enter Choice: 4
```



```
Enter Choice: 4


Finished...

Press Enter to continue ...


Menu
-------
(1) Translation
(2) Rotation
(3) Scaling
(4) Reflection
(5) Shearing
(6) View Figure
(7) Exit

Enter Choice: 5


Finished...

Press Enter to continue ...


Menu
-------
(1) Translation
(2) Rotation
(3) Scaling
(4) Reflection
(5) Shearing
(6) View Figure
(7) Exit

Enter Choice: 6


Choose Projection:
(1) xy-plane
```



```
(7) Exit

Enter Choice: 6


Choose Projection:
(1) xy-plane
(2) xz-plane
(3) yz-plane

Enter Choice: 1
Drawing Original Figure...
1       2       0       2 :: (1, 2) -> (1, 1)
1       1       2.96439e-323    4.94066e-324 :: (1, 1) -> (1, 2.51032e-292)
1       2.51032e-292    0       0 :: (1, 2.51032e-292) -> (1, 2.51032e-292)
1       2.51032e-292    0       1 :: (1, 2.51032e-292) -> (1, 2)

Finished...

Press Enter to continue ...


Menu
-------
(1) Translation
(2) Rotation
(3) Scaling
(4) Reflection
(5) Shearing
(6) View Figure
(7) Exit

Enter Choice: _
```

```
return 0;}
```