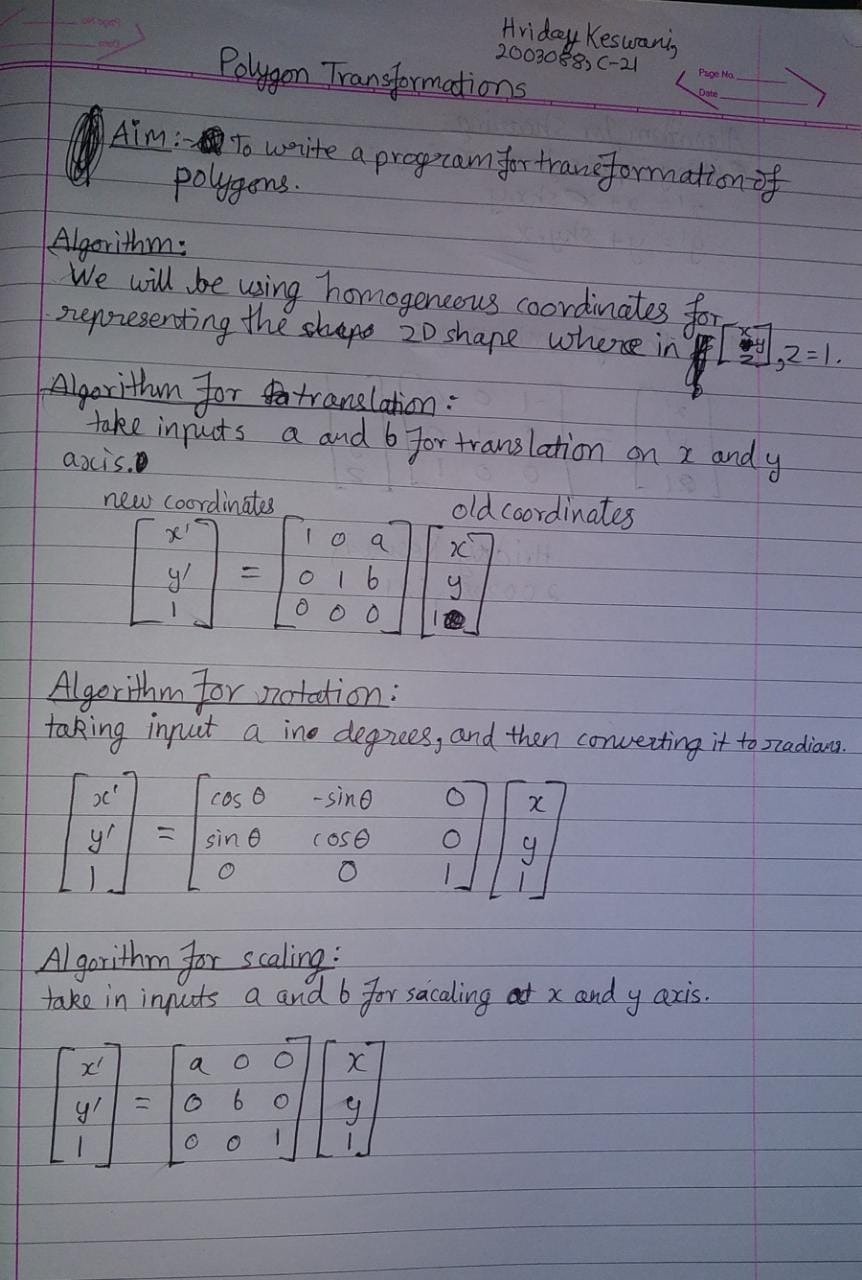
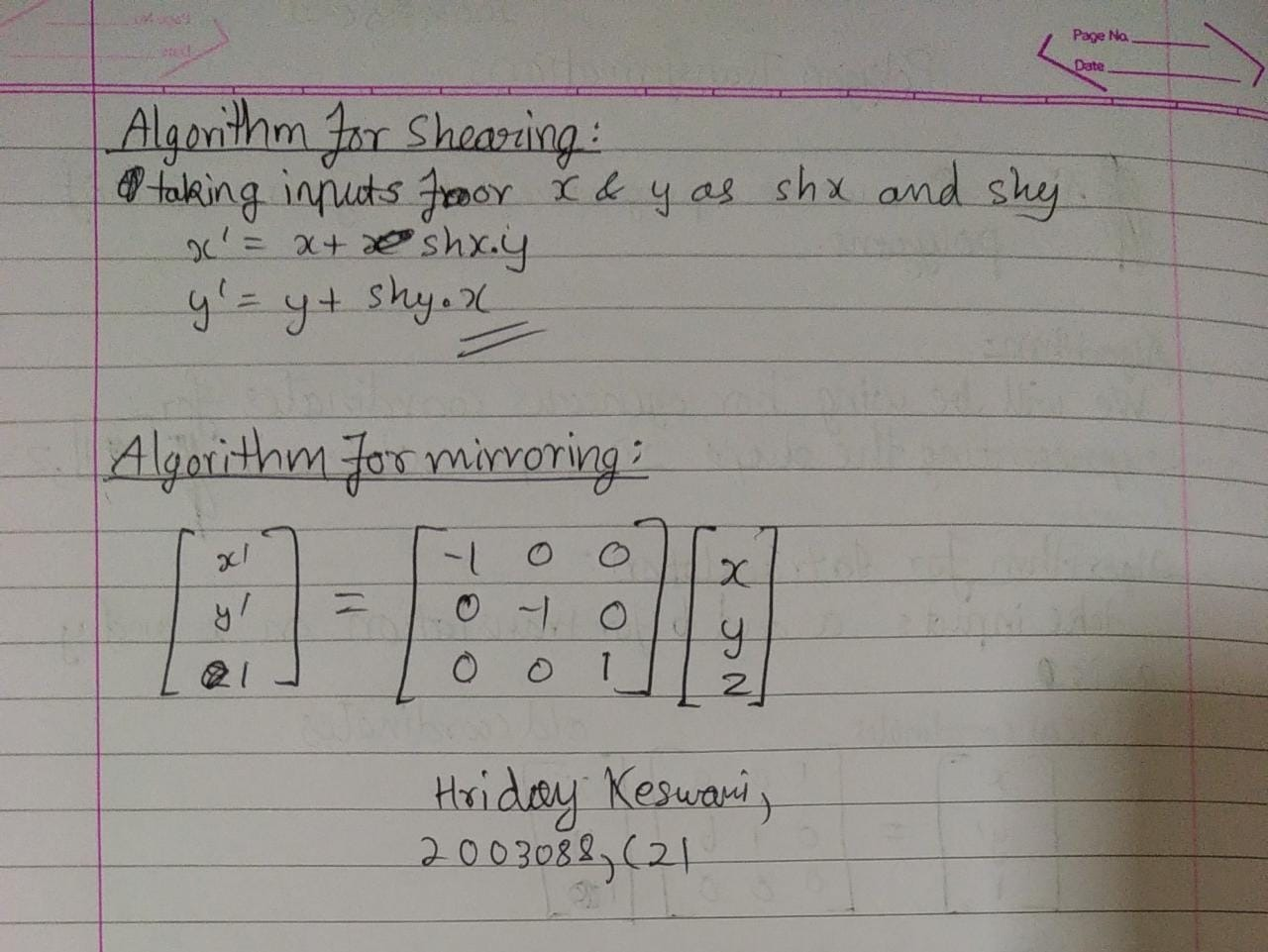
**CG-Assignment**

**Transformations of 2D Polygons**

**Program:**

Writing a program to Transform 2D polygons





**Code:**

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

#include<math.h>

float shape[100][3];

void drawP(int pts, float arr[100][3]);

void addPoint(float x, float y, int index);

void translate(float tx, float ty, int pts, float arr[100][3]);

void rotate(float rot, int pts, float arr[100][3]);

void scale(float sx, float sy, int pts, float arr[100][3]);

void shear(float x, float y, int pts, float arr[100][3]);

void mirror(int pts, float arr[100][3]);

void main(){

int gd = DETECT, gm;

int n,i,choice;

float x,y,a,b;

initgraph(&gd, &gm, "C:\\TURBOC3\\BGI");

line(getmaxx()/2,0,getmaxx()/2,getmaxy());

line(0,getmaxy()/2,getmaxx(),getmaxy()/2);

printf("Enter the number of points the polygon:\n");

scanf("%d",&n);

printf("Enter the coordinates for the polygon\n");

for(i=0;i<n;i++){

scanf("%f%f",&x,&y);

addPoint(x,y,i);

}

drawP(n,shape);

printf("Enter your choice:\n1. Translate\n2. Rotate\n3. Scale\n4. Shear\n5. Mirror\n");

scanf("%d",&choice);

switch(choice){

case 1:

printf("Enter translation for x and y:\n");

scanf("%f%f",&a,&b);

translate(a,b,n,shape);

break;

case 2:

printf("Enter rotaion angle:\n");

scanf("%f",&a);

rotate(a,n,shape);

break;

case 3:

printf("Enter scale for x and y:\n");

scanf("%f%f",&a,&b);

scale(a,b,n,shape);

break;

case 4:

printf("Enter shear for x and y:\n");

scanf("%f%f",&a,&b);

shear(a,b,n,shape);

break;

case 5:

mirror(n,shape);

break;

}

printf("\nHriday Keswani\nC-21\n2003088");

getch();

clrscr();

}

void translate(float tx, float ty, int pts, float arr[100][3]){

int i;

for(i=0;i<pts;i++){

arr[i][0]+=tx;

arr[i][1]+=ty;

}

drawP(pts, shape);

}

void addPoint(float x, float y, int index){

shape[index][0] = x;

shape[index][1] = y;

shape[index][2] = 1;

}

void drawP(int pts, float arr[100][3]){

int i;

for(i=0;i<pts-1;i++){

line((getmaxx()/2)+arr[i][0],((getmaxy()/2)-arr[i][1]),(getmaxx()/2)+arr[i+1][0],((getmaxy()/2)-arr[i+1][1]));

}

line((getmaxx()/2)+arr[0][0],((getmaxy()/2)-arr[0][1]),(getmaxx()/2)+arr[pts-1][0],((getmaxy()/2)-arr[pts-1][1]));

}

void rotate(float rot, int pts, float arr[100][3]){

int i;

rot=rot\*(3.142/180);

for(i=0;i<pts;i++){

arr[i][0]-=sin(rot)\*arr[i][1];

arr[i][1]+=sin(rot)\*arr[i][0];

}

drawP(pts, arr);

}

void scale(float sx, float sy, int pts, float arr[100][3]){

int i;

for(i=0;i<pts;i++){

arr[i][0]\*=sx;

arr[i][1]\*=sy;

}

drawP(pts, arr);

}

void shear(float x, float y, int pts, float arr[100][3]){

int i;

for(i=0;i<pts;i++){

arr[i][0]+=x\*arr[i][1];

arr[i][1]+=y\*arr[i][0];

}

drawP(pts,arr);

}

void mirror(int pts, float arr[100][3]){

int i;

for(i=0;i<pts;i++){

arr[i][0]\*=-1;

arr[i][1]\*=-1;

}

drawP(pts,arr);

}

**Output:**

