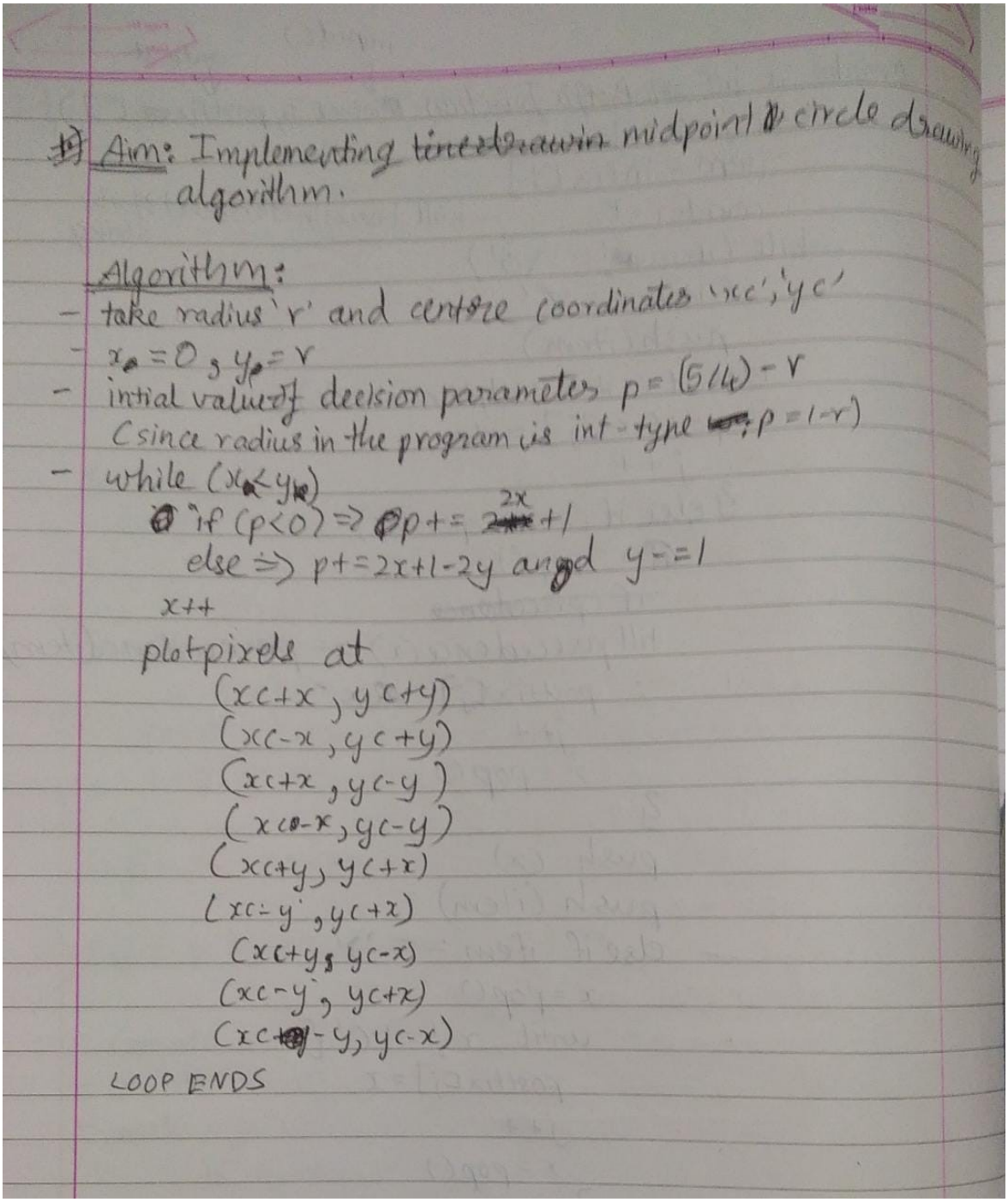
CG-Assignment
Midpoint Circle and Ellipse Algorithms



1) Aim: Implementing ~~three~~ midpoint circle drawing algorithm.

Algorithm:

- take radius 'r' and centre coordinates 'xc', 'yc'
- $x_0 = 0$, $y_0 = r$
- initial value of decision parameter $p = (5/4) - r$
  (since radius in the program is int type ~~or~~ $p = 1 - r$)
- while ($x_0 < y_0$)
  - if ($p < 0$) $\Rightarrow$ $p += 2x + 1$
    else $\Rightarrow$ $p += 2x + 1 - 2y$ and $y -= 1$
  - $x++$
  - plot pixels at
    $(xc+x, yc+y)$
    $(xc-x, yc+y)$
    $(xc+x, yc-y)$
    $(xc-x, yc-y)$
    $(xc+y, yc+x)$
    $(xc-y, yc+x)$
    $(xc+y, yc-x)$
    $(xc-y, yc-x)$

LOOP ENDS

Program:
Drawing circles using midpoint circle algorithm

Code:

```
#include<stdio.h>
#include<graphics.h>
#include<conio.h>

void plotPoints(xc,yc,x,y){
     putpixel(xc+x,yc+y,WHITE);
     putpixel(xc-x,yc+y,WHITE);
     putpixel(xc+x,yc-y,WHITE);
     putpixel(xc-x,yc-y,WHITE);
     putpixel(xc+y,yc+x,WHITE);
     putpixel(xc-y,yc+x,WHITE);
     putpixel(xc+y,yc-x,WHITE);
     putpixel(xc-y,yc-x,WHITE);
}

void mpcircle(int xc, int yc, int r){
     int x,y,p;
     x=0;
     y=r;
     p=1-r;

     plotPoints(xc,yc,x,y);

     while(x<y){
          x++;
          if(p<0){
               p+=2*x+1;
          }else{
               y--;
               p+=2*(x-y)+1;
          }
          plotPoints(xc,yc,x,y);
     }

}


void main(){
     int gd = DETECT,gm;
     int xc,yc,r;
     initgraph(&gd,&gm,"c:\\turboc3\\bgi");
     printf("Enter the cneter coordiantes(x,y)\n");
     scanf("%d%d",&xc,&yc);
     printf("Enter the radius of the circle\n");
     scanf("%d",&r);
```

```
    printf("\nHriday Keswani\nRoll no. 20003088\n");
    mpcircle(xc,yc,r);
    getch();
    closegraph();
}
```

Output:

Program:
Drawing concentric circles midpoint circle algorithm

Code:

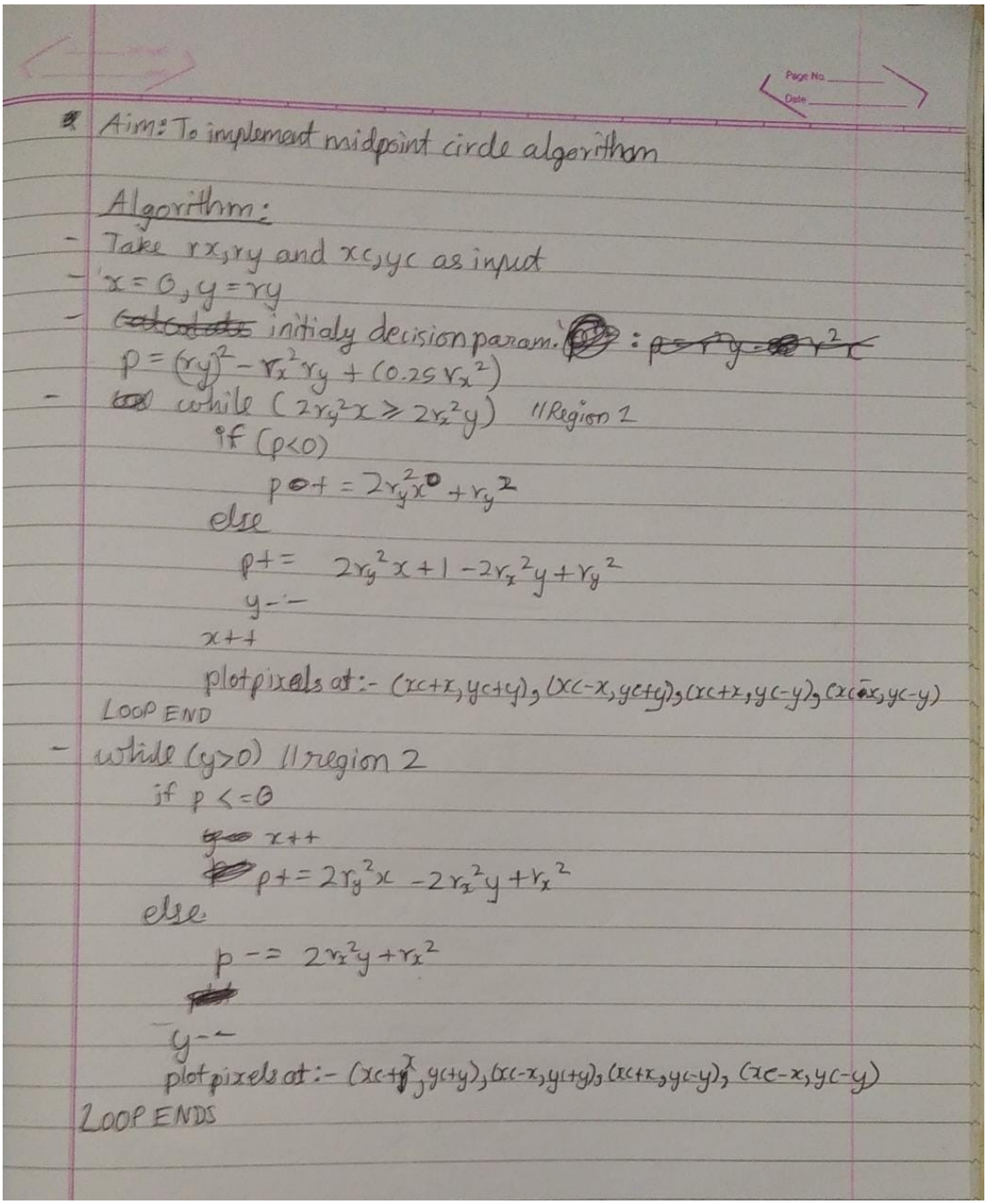
```
#include<stdio.h>
#include<graphics.h>
#include<conio.h>

void plotPoints(xc,yc,x,y){
     putpixel(xc+x,yc+y,WHITE);
     putpixel(xc-x,yc+y,WHITE);
     putpixel(xc+x,yc-y,WHITE);
     putpixel(xc-x,yc-y,WHITE);
     putpixel(xc+y,yc+x,WHITE);
     putpixel(xc-y,yc+x,WHITE);
     putpixel(xc+y,yc-x,WHITE);
     putpixel(xc-y,yc-x,WHITE);
}

void mpcircle(int xc, int yc, int r){
     int x,y,p;
     x=0;
     y=r;
     p=1-r;

     plotPoints(xc,yc,x,y);

     while(x<y){
          x++;
          if(p<0){
               p+=2*x+1;
          }else{
               y--;
               p+=2*(x-y)+1;
          }
          plotPoints(xc,yc,x,y);
     }

     if(r>10){
          r-=10;
          mpcircle(xc,yc,r);
     }

}


void main(){
     int gd = DETECT,gm;
     int xc,yc,r;
```

```
    initgraph(&gd,&gm,"c:\\turboc3\\bgi");
    printf("Enter the center coordiantes(x,y)\n");
    scanf("%d%d",&xc,&yc);
    printf("Enter the radius of the circle\n");
    scanf("%d",&r);
    printf("\nHriday Keswani\nRoll no. 20003088\n");
    mpcircle(xc,yc,r);
    getch();
    closegraph();
}
```

Output:

Program:
Drawing ellipse from midpoint ellipse algorithm

Theory:



Code:

```
#include<stdio.h>
#include<graphics.h>
#include<conio.h>
```

```
void main(){
     int gd = DETECT,gm;
     int xc,yc,x,y;
     long rx,ry;
     float p;

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

     printf("Enter the center coordinates:\n");
     scanf("%d%d",&xc,&yc);
     printf("Enter Rx and Ry:\n");
     scanf("%ld%ld",&rx,&ry);
     printf("\nHriday Keswani\nRoll no. 20003088\n");

     p = ry*ry-(rx*rx*ry)+(0.25*rx*rx);
     x=0;
     y=ry;

     while(2.0*ry*ry*x <= 2.0*rx*rx*y){
          if(p<0){
               x++;
               p = p+2*ry*ry*x+ry*ry;
          }else{
               x++;
               y--;
               p = p+2*ry*ry-2*rx*rx*y-ry*ry;
          }
          putpixel(xc+x,yc+y,WHITE);
          putpixel(xc-x,yc+y,WHITE);
          putpixel(xc+x,yc-y,WHITE);
          putpixel(xc-x,yc-y,WHITE);
     }

     p = ry*ry*(x+0.5)+rx*rx*(y-1)*(y-1)-rx*rx*ry*ry;

     while(y>0){
          if(p<=0){
               x++;
               y--;
               p = p+2*ry*ry*x-2*rx*rx*y+rx*rx;
          }else{
               y--;
               p=p-2*rx*rx*y+rx*rx;
          }
          putpixel(xc+x,yc+y,WHITE);
          putpixel(xc-x,yc+y,WHITE);
          putpixel(xc+x,yc-y,WHITE);
          putpixel(xc-x,yc-y,WHITE);
     }
```

```
    getch();
    closegraph();
}
```

Output: