```
#include"graphics.h"
#include<string>
#include<iostream>

using namespace std;
int first_x = 0;
int first_y = 0;

class shapes
{



public:

	int* points = new int[100];

	virtual void Draw() = 0;
	//	virtual void Delete() = 0;



};

class colours
{

private:
	int selected_colour;
	bool bucket_enable;
public:
```

```
int bucket_x, bucket_y;

void check_colour(int x1, int y1, char ch)
{

	if ((x1 >= 5 && x1 <= 50) && (y1 >= 14 && y1 <= 49))
	{

		selected_colour = 15;

	}



	else if ((x1 >= 65 && x1 <= 109) && (y1 >= 14 && y1 <= 49))
	{

		selected_colour = 11;

	}
	//////////////////////////////////////////
	else	if ((x1 >= 5 && x1 <= 50) && (y1 >= 54 && y1 <= 89))
	{

		selected_colour = 14;

	}
	else if ((x1 >= 65 && x1 <= 108) && (y1 >= 54 && y1 <= 89))
```

```
{

        selected_colour = 6;

}
////////////////////////////////////////////////////////
else if ((x1 >= 5 && x1 <= 50) && (y1 >= 95 && y1 <= 128))
{

        selected_colour = 4;

}
else if ((x1 >= 65 && x1 <= 108) && (y1 >= 95 && y1 <= 128))
{

        selected_colour = 3;

}
////////////////////////////////////////////////////////////////
else if ((x1 >= 5 && x1 <= 50) && (y1 >= 136 && y1 <= 169))
{

        selected_colour = 10;

}
else if ((x1 >= 65 && x1 <= 108) && (y1 >= 136 && y1 <= 169))
{

        selected_colour = 1;
```

```
}
//////////////////////////////////////////////////////////
else if ((x1 >= 5 && x1 <= 50) && (y1 >= 175 && y1 <= 210))
{

	selected_colour = 7;

}
else if ((x1 >= 65 && x1 <= 110) && (y1 >= 175 && y1 <= 209))
{

	selected_colour = 0;

}



if (ch == 'M')
{
	while (!ismouseclick(WM_LBUTTONDOWN))
	{
	}

	getmouseclick(WM_LBUTTONDOWN, bucket_x, bucket_y);



	isbucket_enable(bucket_x, bucket_y, 'M');
}
```

```
}

void isbucket_enable(int x, int y, char ch)
{
	if ((x >= 1 && x <= 60) && (y >= 575 && y <= 635))
	{
		bucket_enable = 1;
	}
	else
		bucket_enable = 0;

	if (bucket_enable == 1)
	{
		if (ch == 'M')
		{

			fill_shape(selected_colour, bucket_enable);



		}







	}
}

bool get_bucket_Status()
{
```

```
	return bucket_enable;

}

int get_Selectedcolour()
{

	return selected_colour;
}

void fill_shape(int colour, bool bucket)
{

	if (bucket == 1)
	{
		int colr_x1 = 0; int colr_y1 = 0;



		while (!ismouseclick(WM_LBUTTONDOWN))
		{
		}

		getmouseclick(WM_LBUTTONDOWN, colr_x1, colr_y1);

		setcolor(BLACK);
		// circle(x1, y1, radius);

		setfillstyle(SOLID_FILL, colour);
```

```
                floodfill(colr_x1, colr_y1, BLACK);

            }
        }
};

class CirclE :public shapes, public colours
{

private:
        int circle_colour;
        int circle_bucket;
        int radius;
public:

        void Draw()

        {
            int  x1 = 0, y1 = 0, x2 = 0, y2 = 0;
            float  distance_1 = 0, distance_2 = 0, distance_3 = 0;
            int selected_colour;
            int color_x, color_y, bucket_x, bucket_y;


            while (!ismouseclick(WM_LBUTTONDOWN))
            {
            }

            getmouseclick(WM_LBUTTONDOWN, x1, y1);
```

```
cout << x1 << " , ";
cout << y1 << endl;
clearmouseclick(WM_LBUTTONDOWN);

while (!ismouseclick(WM_LBUTTONDOWN))
{
}
getmouseclick(WM_LBUTTONDOWN, x2, y2);

cout << x2 << ", ";
cout << y2 << endl;


clearmouseclick(WM_LBUTTONDOWN);


distance_1 = (x2 - x1);
distance_1 = pow(distance_1, 2);

distance_2 = (y2 - y1);
distance_2 = pow(distance_2, 2);

distance_3 = distance_1 + distance_2;
distance_3 = pow(distance_3, 0.5);

radius = distance_3;

setcolor(BLACK);
```

```
circle(x1, y1, radius);



while (!ismouseclick(WM_LBUTTONDOWN))
{
}
getmouseclick(WM_LBUTTONDOWN, color_x, color_y);



if (color_y >= 0 && color_y <= 223)
{

        check_colour(color_x, color_y, 'c');

        selected_colour = get_Selectedcolour();




        while (!ismouseclick(WM_LBUTTONDOWN))
        {
        }

        getmouseclick(WM_LBUTTONDOWN, bucket_x, bucket_y);




        isbucket_enable(bucket_x, bucket_y, 'c');
        circle_bucket = get_bucket_Status();
```

```
                        fill_shape(selected_colour, circle_bucket);

                }
        }

};

class RectAngle :public shapes, public colours
{

private:

        int selected_colour;
        bool Rectangle_bucket;
public:



        void Draw()
        {
                int  x1 = 0, y1 = 0, x2 = 0, y2 = 0;

                int color_x, color_y, bucket_x, bucket_y;
                float  distance_1 = 0, distance_2 = 0, distance_3 = 0;



                while (!ismouseclick(WM_LBUTTONDOWN))
```

```
{
}

getmouseclick(WM_LBUTTONDOWN, x1, y1);

cout << x1 << " , ";
cout << y1 << endl;
clearmouseclick(WM_LBUTTONDOWN);

while (!ismouseclick(WM_LBUTTONDOWN))
{
}
getmouseclick(WM_LBUTTONDOWN, x2, y2);

cout << x2 << ", ";
cout << y2 << endl;


clearmouseclick(WM_LBUTTONDOWN);


setcolor(BLACK);
rectangle(x1, y1, x2, y2);
rectangle(x2, y2, x1, y1);



while (!ismouseclick(WM_LBUTTONDOWN))
{
```

```
}
getmouseclick(WM_LBUTTONDOWN, color_x, color_y);


if (color_y >= 0 && color_y <= 223)
{

        check_colour(color_x, color_y, 'r');

        selected_colour = get_Selectedcolour();



        while (!ismouseclick(WM_LBUTTONDOWN))
        {
        }

        getmouseclick(WM_LBUTTONDOWN, bucket_x, bucket_y);



        isbucket_enable(bucket_x, bucket_y, 'r');
        Rectangle_bucket = get_bucket_Status();


        fill_shape(selected_colour, Rectangle_bucket);

}
```

```
        }

};

class polygon :public shapes, public colours
{

private:
        int selected_colour;
        bool Square_bucket;
public:

        int color_x, color_y, bucket_x, bucket_y;

        void Draw()
        {
                int  x1 = 0, y1;
                int count = 0;
                int arr[20];
                int i = 0;

                while (!ismouseclick(WM_LBUTTONDBLCLK))
                {

                        while (!ismouseclick(WM_LBUTTONDOWN))
                        {
                        }
```

```
getmouseclick(WM_LBUTTONDOWN, x1, y1);

cout << x1 << " , ";
cout << y1 << endl;
clearmouseclick(WM_LBUTTONDOWN);

while (!ismouseclick(WM_LBUTTONDOWN))
{
}

arr[i] = x1;
arr[i + 1] = y1;

if (count == 0)
{
        first_x = x1;
        first_y = y1;
}



i++;
count++;



if ((ismouseclick(WM_LBUTTONDOWN)))
{
```

```
                arr[count] = first_x;
                arr[count + 1] = first_y;

                break;
        }

        clearmouseclick(WM_RBUTTONDOWN);
}

while (!ismouseclick(WM_LBUTTONDOWN))
{
}
getmouseclick(WM_LBUTTONDOWN, color_x, color_y);

if (color_y >= 0 && color_y <= 223)
{
        check_colour(color_x, color_y, 'e');

        selected_colour = get_Selectedcolour();

        while (!ismouseclick(WM_LBUTTONDOWN))
        {
        }
```

```
                    getmouseclick(WM_LBUTTONDOWN, bucket_x, bucket_y);



                    isbucket_enable(bucket_x, bucket_y, 'e');
                    Square_bucket = get_bucket_Status();


                    fill_shape(selected_colour, Square_bucket);

                }

        }
};

class EllipsE :public shapes, public colours
{

private:
        int selected_colour;
        bool ellipse_bucket;
public:


        void Draw()
        {
                int  center_x = 0, center_y = 0, radius_x1 = 0, radius_x2 = 0, radius_y1 = 0, radius_y2 = 0;

                float  distance_1 = 0, distance_2 = 0, distance_3 = 0;
```

```
int color_x, color_y, bucket_x, bucket_y;

while (!ismouseclick(WM_LBUTTONDOWN))
{
}
getmouseclick(WM_LBUTTONDOWN, center_x, center_y);
clearmouseclick(WM_LBUTTONDOWN);
////////////////////////////////////////////////////////////////////////////////

while (!ismouseclick(WM_LBUTTONDOWN))
{
}
getmouseclick(WM_LBUTTONDOWN, radius_x1, radius_x2);
clearmouseclick(WM_LBUTTONDOWN);

////////////////////////////////////////////////////////////

while (!ismouseclick(WM_LBUTTONDOWN))
{
}
getmouseclick(WM_LBUTTONDOWN, radius_y1, radius_y2);
clearmouseclick(WM_LBUTTONDOWN);




setcolor(BLACK);
ellipse(center_x, center_y, 0, 360, radius_x1 / 10, radius_y2 / 10);
```

```
while (!ismouseclick(WM_LBUTTONDOWN))
{
}
getmouseclick(WM_LBUTTONDOWN, color_x, color_y);


if (color_y >= 0 && color_y <= 223)
{
        check_colour(color_x, color_y, 'e');

        selected_colour = get_Selectedcolour();



        while (!ismouseclick(WM_LBUTTONDOWN))
        {
        }

        getmouseclick(WM_LBUTTONDOWN, bucket_x, bucket_y);



        isbucket_enable(bucket_x, bucket_y, 'e');
        ellipse_bucket = get_bucket_Status();
```

```
                        fill_shape(selected_colour, ellipse_bucket);

                }









        }

};

class LinE :public shapes, public colours
{

private:
        int selected_colour;
        bool line_bucket;

public:

        int color_x, color_y, bucket_x, bucket_y;



        void Draw()
        {
                int  x1 = 0, y1 = 0, x2 = 0, y2 = 0;
```

```
while (!ismouseclick(WM_LBUTTONDOWN))
{
}

getmouseclick(WM_LBUTTONDOWN, x1, y1);
clearmouseclick(WM_LBUTTONDOWN);


while (!ismouseclick(WM_LBUTTONDOWN))
{
}

getmouseclick(WM_LBUTTONDOWN, x2, y2);
clearmouseclick(WM_LBUTTONDOWN);



setcolor(BLACK);
line(x1, y1, x2, y2);



while (!ismouseclick(WM_LBUTTONDOWN))
{
}
getmouseclick(WM_LBUTTONDOWN, color_x, color_y);
```

```
        if (color_y >= 0 && color_y <= 223)
        {

            check_colour(color_x, color_y, 'l');

            selected_colour = get_Selectedcolour();



            while (!ismouseclick(WM_LBUTTONDOWN))
            {
            }

            getmouseclick(WM_LBUTTONDOWN, bucket_x, bucket_y);



            isbucket_enable(bucket_x, bucket_y, 'l');
            line_bucket = get_bucket_Status();

            fill_shape(selected_colour, line_bucket);

        }


}
```

```
};

class TrianglE :public shapes, public colours
{
private:
	int selected_colour;
	bool Triangle_bucket;

public:

	int color_x, color_y, bucket_x, bucket_y;




	void Draw()
	{
		int  point1_x1 = 0, point1_y1 = 0, point2_x1 = 0, point2_y1 = 0, point3_x1 = 0, point3_y1
= 0;



		while (!ismouseclick(WM_LBUTTONDOWN))
		{
		}
		getmouseclick(WM_LBUTTONDOWN, point1_x1, point1_y1);
		clearmouseclick(WM_LBUTTONDOWN);
		////////////////////////////////////////////////////////////////////////////////
```

```
while (!ismouseclick(WM_LBUTTONDOWN))
{
}
getmouseclick(WM_LBUTTONDOWN, point2_x1, point2_y1);
clearmouseclick(WM_LBUTTONDOWN);

//////////////////////////////////////////////////////////

while (!ismouseclick(WM_LBUTTONDOWN))
{
}
getmouseclick(WM_LBUTTONDOWN, point3_x1, point3_y1);
clearmouseclick(WM_LBUTTONDOWN);


setcolor(BLACK);
line(point1_x1, point1_y1, point2_x1, point2_y1);
setcolor(BLACK);
line(point2_x1, point2_y1, point3_x1, point3_y1);
setcolor(BLACK);
line(point3_x1, point3_y1, point1_x1, point1_y1);



while (!ismouseclick(WM_LBUTTONDOWN))
{
}
getmouseclick(WM_LBUTTONDOWN, color_x, color_y);
```

```
if (color_y >= 0 && color_y <= 223)
{

        check_colour(color_x, color_y, 't');

        selected_colour = get_Selectedcolour();



        while (!ismouseclick(WM_LBUTTONDOWN))
        {
        }

        getmouseclick(WM_LBUTTONDOWN, bucket_x, bucket_y);



        isbucket_enable(bucket_x, bucket_y, 't');
        Triangle_bucket = get_bucket_Status();


        fill_shape(selected_colour, Triangle_bucket);

}
```

```
        }

};

class pencil :public shapes
{

public:
        void Draw()
        {

                int x1, y1;

                while (!ismouseclick(WM_LBUTTONDOWN))
                {
                }

                clearmouseclick(WM_LBUTTONDOWN);

                setcolor(0);

                while (1)
                {

                        if (ismouseclick(WM_LBUTTONDOWN))
                        {
                                clearmouseclick(WM_LBUTTONDOWN);
```

```
                    break;
                    cout << "0K'";
                }

                while (ismouseclick(WM_LBUTTONUP))
                {
                    while (ismouseclick(WM_MOUSEMOVE))
                    {
                        getmouseclick(WM_MOUSEMOVE, x1, y1);
                        clearmouseclick(WM_MOUSEMOVE);
                        setfillstyle(11, 0);
                        bar(x1, y1, x1 + 5, y1 + 5);

                    }
                }
            }
        }


};

class Eraser :public shapes
{

public:
    void Draw()
    {

        int x1, y1;
```

```
while (!ismouseclick(WM_LBUTTONDOWN))
{
}

clearmouseclick(WM_LBUTTONDOWN);

setcolor(15);

while (1)
{
	setcolor(15);

	if (ismouseclick(WM_LBUTTONDOWN))
	{
		clearmouseclick(WM_LBUTTONDBLCLK);
		break;
		cout << "0K'";
	}

	while (ismouseclick(WM_LBUTTONUP))
	{
		while (ismouseclick(WM_MOUSEMOVE))
		{

			getmouseclick(WM_MOUSEMOVE, x1, y1);
			clearmouseclick(WM_MOUSEMOVE);
			setfillstyle(1, 15);
			bar(x1, y1, x1 + 10, y1 + 10);
```

```
                                        }
                                }

                        }


                }


};

class FileOperation :public colours
{

public:

        void save_file()
        {

                writeimagefile("paint.bmp", 200, 0, 1400, 750, 0, (HWND)0);

        }

        void Loading_file()
        {
                readimagefile("paint.bmp", 200, 0, 1400, 750);
        }
```

```
};



void display()
{
        //      floodfill(0,0,2);

        setbkcolor(MAGENTA);


        //setcolor(GREEN);
//setfillstyle(1,GREEN);
 //bar(0,0,200,1000);


// selection panel

        setcolor(MAGENTA);
        setfillstyle(SOLID_FILL, MAGENTA);
        rectangle(0, 0, 200, 1000);
        floodfill(20, 50, MAGENTA);
```

```
//First Row:


setcolor(WHITE);

setfillstyle(SOLID_FILL, WHITE);

rectangle(5, 15, 50, 50);

floodfill(21, 31, WHITE);


setcolor(WHITE);
setfillstyle(SOLID_FILL, LIGHTCYAN);
rectangle(110, 15, 65, 50);
floodfill(86, 36, WHITE);


setcolor(WHITE);
setfillstyle(SOLID_FILL, WHITE);
rectangle(201, 1, 1355, 736);
floodfill(530, 258, WHITE);

//Second Row:


setcolor(WHITE);
setfillstyle(SOLID_FILL, YELLOW);
```

```
rectangle(5, 90, 50, 54);
floodfill(10, 60, WHITE);


setcolor(WHITE);
setfillstyle(SOLID_FILL, BROWN);
rectangle(110, 90, 65, 54);
floodfill(93, 72, WHITE);



//Third Row:
setcolor(WHITE);
setfillstyle(SOLID_FILL, RED);
rectangle(5, 130, 50, 95);
floodfill(21, 117, WHITE);

setcolor(WHITE);
setfillstyle(SOLID_FILL, CYAN);
rectangle(110, 130, 65, 95);
floodfill(93, 110, WHITE);


//Fourth Row:
setcolor(WHITE);
setfillstyle(SOLID_FILL, 10);
rectangle(5, 170, 50, 135);
```

```
floodfill(29, 149, WHITE);

setcolor(WHITE);
setfillstyle(SOLID_FILL, BLUE);
rectangle(110, 170, 65, 135);
floodfill(91, 159, WHITE);

//Fifth Row:
setcolor(WHITE);
setfillstyle(SOLID_FILL, LIGHTGRAY);
rectangle(5, 210, 50, 175);
floodfill(23, 198, WHITE);



setcolor(WHITE);
setfillstyle(SOLID_FILL, BLACK);
rectangle(110, 210, 65, 175);
floodfill(75, 198, WHITE);

//Sixth Row:

circle(25, 250, 25);
setfillstyle(SOLID_FILL, LIGHTCYAN);
floodfill(25, 250, 15);
outtextxy(6, 292, "Circle:");


setcolor(WHITE);
```

```
setfillstyle(SOLID_FILL, LIGHTCYAN);
rectangle(85, 231, 186, 272);
floodfill(127, 253, WHITE);
outtextxy(98, 294, "Rectangle:");




//Seventh Row:
int arr[] = { 44, 323,  1, 379, 91, 379,   44, 323 };



setcolor(WHITE);
drawpoly(4, arr);
fillpoly(4, arr);
setfillstyle(SOLID_FILL, LIGHTCYAN);
floodfill(25, 250, WHITE);
outtextxy(17, 386, "Triangle:");



setcolor(WHITE);
setfillstyle(SOLID_FILL, LIGHTCYAN);
int Arr[] = { 113,345,   113,380, 180,380,  180,342,  149,317, 113,345 };
drawpoly(6, Arr);
//rectangle(114, 325, 181, 372);
floodfill(144, 347, WHITE);
outtextxy(119, 389, "polygon:");
```

```
setcolor(WHITE);
setfillstyle(SOLID_FILL, LIGHTCYAN);
ellipse(81, 466, 0, 360, 80, 40);
fillellipse(81, 466, 80, 40);
outtextxy(16, 510, "Ellipse:");


setcolor(WHITE);
setfillstyle(SOLID_FILL, LIGHTCYAN);
rectangle(12, 536, 154, 540);
floodfill(75, 538, WHITE);
outtextxy(14, 546, "Line:");


/*
        outtextxy(14, 608, "Bucket:");
        outtextxy(136, 609, "Delete:");
        outtextxy(15, 646, "Save:");
        outtextxy(135, 649, "Undo:");
        outtextxy(75, 677, "Redo:");

        */

setcolor(BLACKNESS);

readimagefile("bucket.jpg", 1, 575, 60, 635);
readimagefile("erase.jpg", 68, 575, 127, 635);
```

```
readimagefile("pen.jpg", 135, 575, 194, 635);

readimagefile("save.jpg", 1, 644, 60, 704);
readimagefile("load.jpg", 68, 644, 127, 704);
readimagefile("redo.jpg", 135, 644, 194, 704);




//fillellipse(130, 450, 30, 20);




//line(60, 350, 90, 300);

//Eighth Row:
/*
int Arr[] = { 20, 500 ,     70, 500,      30,400, 40,450,    60,350,20,500 };
drawpoly(4, Arr);
fillpoly(4, Arr);



ellipse(130, 450, 0, 360, 30, 20);
fillellipse(130, 450, 30, 20);
```

```
        outtextxy(20, 550, "Delete");


        outtextxy(20, 600, "Bucket");

*/


/*
int x1 = 0, y1 = 0, x2 = 0, y2 = 0;

while (!ismouseclick(WM_RBUTTONDOWN))
{

        while (!ismouseclick(WM_LBUTTONDOWN))
        {
        }

        ismouseclick(ismouseclick(WM_LBUTTONDOWN));

        getmouseclick(WM_LBUTTONDOWN, x1, y1);

        cout << x1 << " , ";
        cout << y1 << endl;
        clearmouseclick(WM_LBUTTONDOWN);
```

```
        while (!ismouseclick(WM_LBUTTONDOWN))
        {
        }
        getmouseclick(WM_LBUTTONDOWN, x2, y2);

        cout << x2 << ", ";
        cout << y2 << endl;


        clearmouseclick(WM_LBUTTONDOWN);
}


*/
}


int main()
{
        initwindow(1400, 750, "PAINT");

        display();

        shapes* drawing[7];
        int x1 = 0, y1 = 0, x2 = 0, y2 = 0;

        while (!ismouseclick(WM_RBUTTONDOWN))
        {
```

```
clearmouseclick(WM_LBUTTONDBLCLK);
clearmouseclick(WM_LBUTTONDOWN);
clearmouseclick(WM_LBUTTONUP);
clearmouseclick(WM_RBUTTONDOWN);
clearmouseclick(WM_RBUTTONUP);



while (!ismouseclick(WM_LBUTTONDOWN))
{
}

getmouseclick(WM_LBUTTONDOWN, x1, y1);

cout << x1 << " , ";
cout << y1 << endl;

clearmouseclick(WM_LBUTTONDOWN);



if ((x1 >= 1 && x1 <= 49) && (y1 >= 225 && y1 <= 274))
{

        setcolor(BLACK);
        setfillstyle(SOLID_FILL, GREEN);
        circle(25, 230, 4);
        floodfill(25, 230, BLACK);
```

```
            drawing[0] = new CirclE;
            drawing[0]->Draw();
}

else if ((x1 >= 85 && x1 <= 186) && (y1 >= 231 && y1 <= 271))

{

            setcolor(BLACK);
            setfillstyle(SOLID_FILL, GREEN);
            circle(100, 250, 4);
            floodfill(100, 250, BLACK);

            drawing[1] = new RectAngle;

            drawing[1]->Draw();



}

else if ((x1 >= 114 && x1 <= 181) && (y1 >= 325 && y1 <= 372))

{

            setcolor(BLACK);
            setfillstyle(SOLID_FILL, GREEN);
            circle(150, 360, 4);
            floodfill(150, 360, BLACK);
```

```
            drawing[2] = new polygon;
            drawing[2]->Draw();
}

else if ((x1 >= 0 && x1 <= 160) && (y1 >= 426 && y1 <= 505))

{
            setcolor(BLACK);
            setfillstyle(SOLID_FILL, GREEN);
            circle(100, 450, 4);
            floodfill(100, 450, BLACK);

            drawing[3] = new EllipsE;
            drawing[3]->Draw();
}

else if ((x1 >= 18 && x1 <= 165) && (y1 >= 560 && y1 <= 568))

{

            setcolor(BLACK);
            setfillstyle(SOLID_FILL, GREEN);
            circle(100, 563, 4);
            floodfill(100, 563, BLACK);

            drawing[4] = new LinE;
            drawing[4]->Draw();

}
```

```
		else if (((x1 >= 0 && x1 <= 91) || ((x1 >= 23 && x1 <= 68) || (x1 >= 33 && x1 <= 56))) &&
(y1 >= 324 && y1 <= 378))

		{
			setcolor(BLACK);
			setfillstyle(SOLID_FILL, GREEN);
			circle(50, 350, 4);
			floodfill(50, 350, BLACK);

			drawing[5] = new TrianglE;
			drawing[5]->Draw();

		}

		else if (y1 >= 0 && y1 <= 223)
		{



			colours colr;
			colr.check_colour(x1, y1, 'M');

		}

		else if ((x1 >= 135 && x1 <= 194) && (y1 >= 575 && y1 <= 635))
		{
			setcolor(BLACK);
			setfillstyle(SOLID_FILL, GREEN);
```

```
            circle(150, 600, 4);
            floodfill(150, 600, BLACK);

            pencil pen;
            pen.Draw();

}

else if ((x1 >= 68 && x1 <= 127) && (y1 >= 574 && y1 <= 634))
{

            setcolor(BLACK);
            setfillstyle(SOLID_FILL, GREEN);
            circle(100, 600, 4);
            floodfill(100, 600, BLACK);

            Eraser e;
            e.Draw();
}

else if ((x1 >= 1 && x1 <= 60) && (y1 >= 644 && y1 <= 700))
{

            setcolor(BLACK);
            setfillstyle(SOLID_FILL, GREEN);
            circle(50, 650, 4);
            floodfill(50, 650, BLACK);

            FileOperation S;
```

```
            S.save_file();

}

else if ((x1 >= 67 && x1 <= 127) && (y1 >= 644 && y1 <= 700))
{
            setcolor(BLACK);
            setfillstyle(SOLID_FILL, GREEN);
            circle(100, 650, 4);
            floodfill(100, 650, BLACK);

            FileOperation Load;

            Load.Loading_file();
}

else if ((x1 >= 134 && x1 <= 194) && (y1 >= 644 && y1 <= 700))
{

            setcolor(BLACK);
            setfillstyle(SOLID_FILL, GREEN);
            circle(100, 650, 4);
            floodfill(100, 650, BLACK);

            FileOperation undo;

            undo.Loading_file();
```

```
                }

        }



        system("pause");

}
```