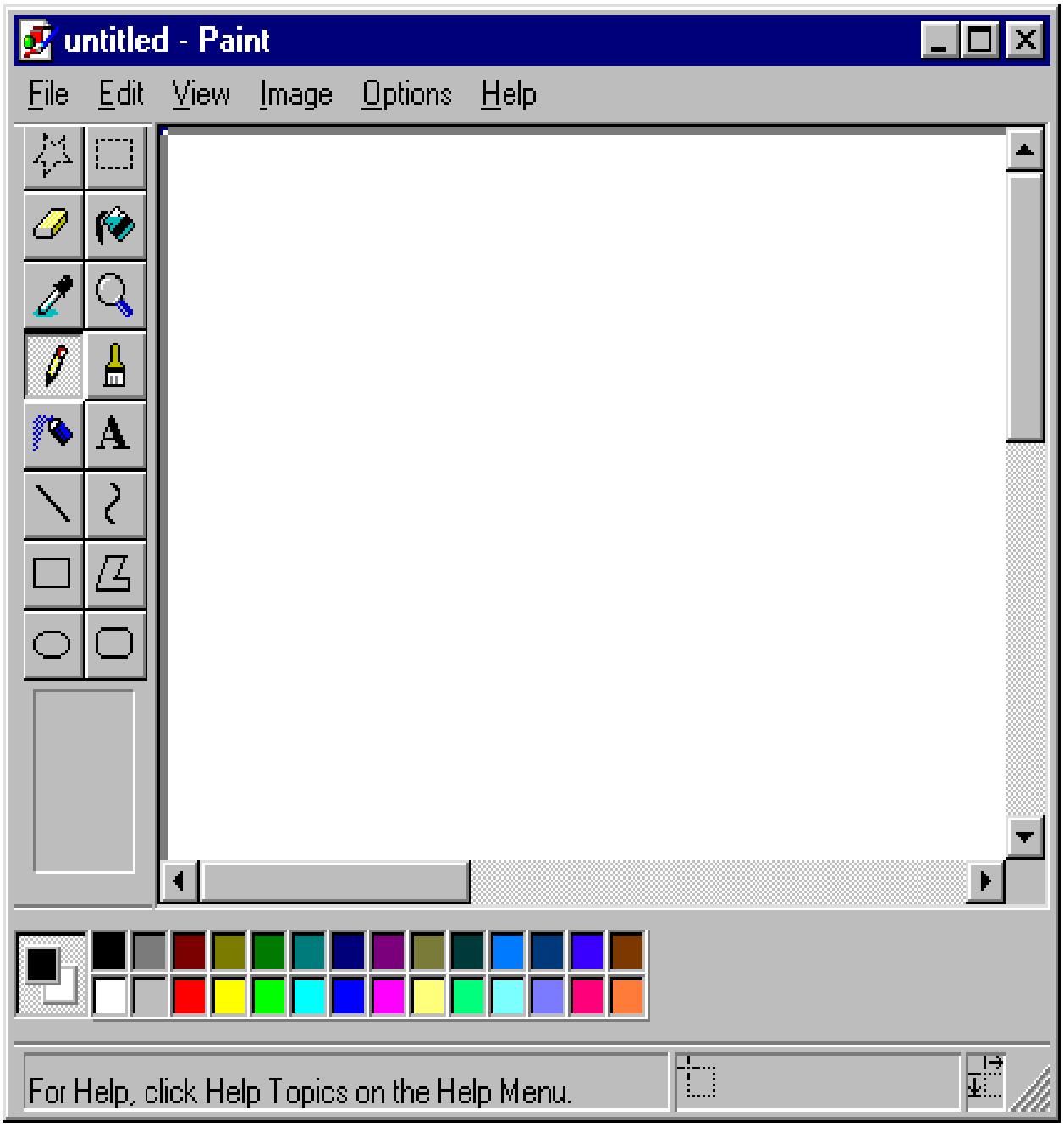
Paint

By:Kovi Szental,Yeshivat Horev



Contents

Introduction : 3
Work Space: 3
Functions provided to us by Teachers: 3
User Manual: 4
Screen Layout: 4
Short keys: 5
Saving and Opening Files: 5
Some Things you can draw: 6
Program build: 7
Program Flow Chart: 7
Function Call Tree: 8
Programs Main Functions: 9
Set_Pixel: 9
Bmp_File: 9
Get_Key_Press: 9
Mouse: 9
Help_Menu_In: 9
Get_Color: 9
DrawLine: 9
Pencil: 9
Eraser: 9
DrawSquare: 9
Bucket_Fill: 9
Save_Pic: 9
Open_Pic: 9
Main Program Variables and File Strings: 10
Color: 10
Mode: 10
Point1x, Point2x,Point1y, Point2y: 10
FileHandle: 10
Intro.Bmp: 10
Help.Bmp, Help2.Bmp: 10
Draw.bmp: 10
Colorp.Bmp: 10
LastSave.bmp: 10
Algorithms: 11
Bressenham Line Algorithm: 11
Bucket Fill Algorithm: 11
Problems and Solutions 12
Square,Get Color ,Eraser : 12
Bucket_Fill: 12
Saving Photo : 12
Extra Additions to the Program: 13
Circle: 13
Picture: 13
Line Thickness: 13
Colour 2: 13
SprayPaint: 13
Thank You: 13

Introduction :

The program I chose to design is Paint. I thought it would be fun and challenging to work with graphics in Assembly (which it was).

Work Space:

The Work Space I used was DosBox, which is an Emulator Program that Emulates an x86 processer including an operating system compatible with Dos. The development is done by Turbo Assembler, or TASM for short. I also use basic input/output functions provided by Gvahim.

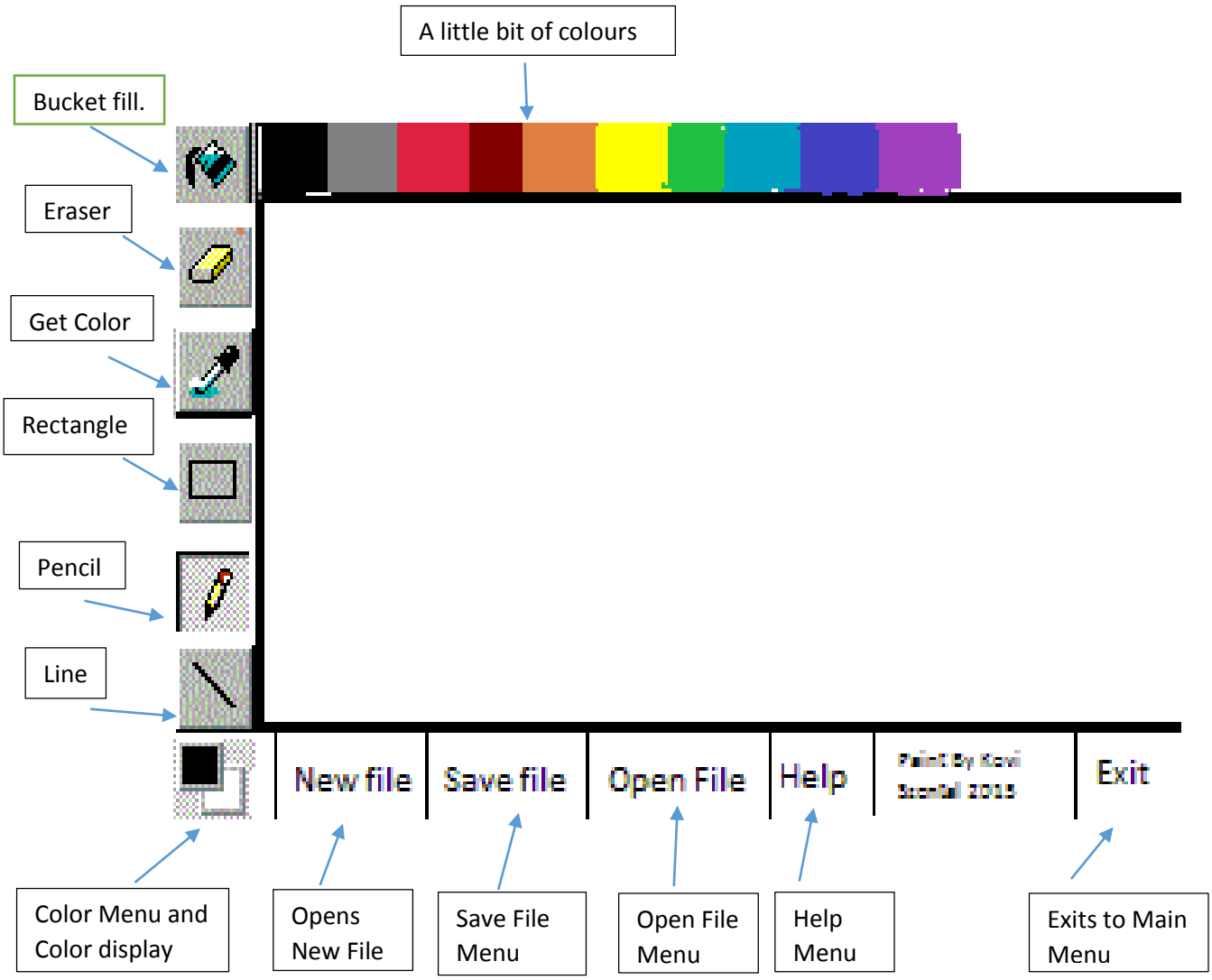
Functions provided to us by Teachers:

The functions inside the file Gvahim.Asm were provided by my teachers.
Out of the Functions provided to us I used:
Print_Str, Scan_Str, New_Line

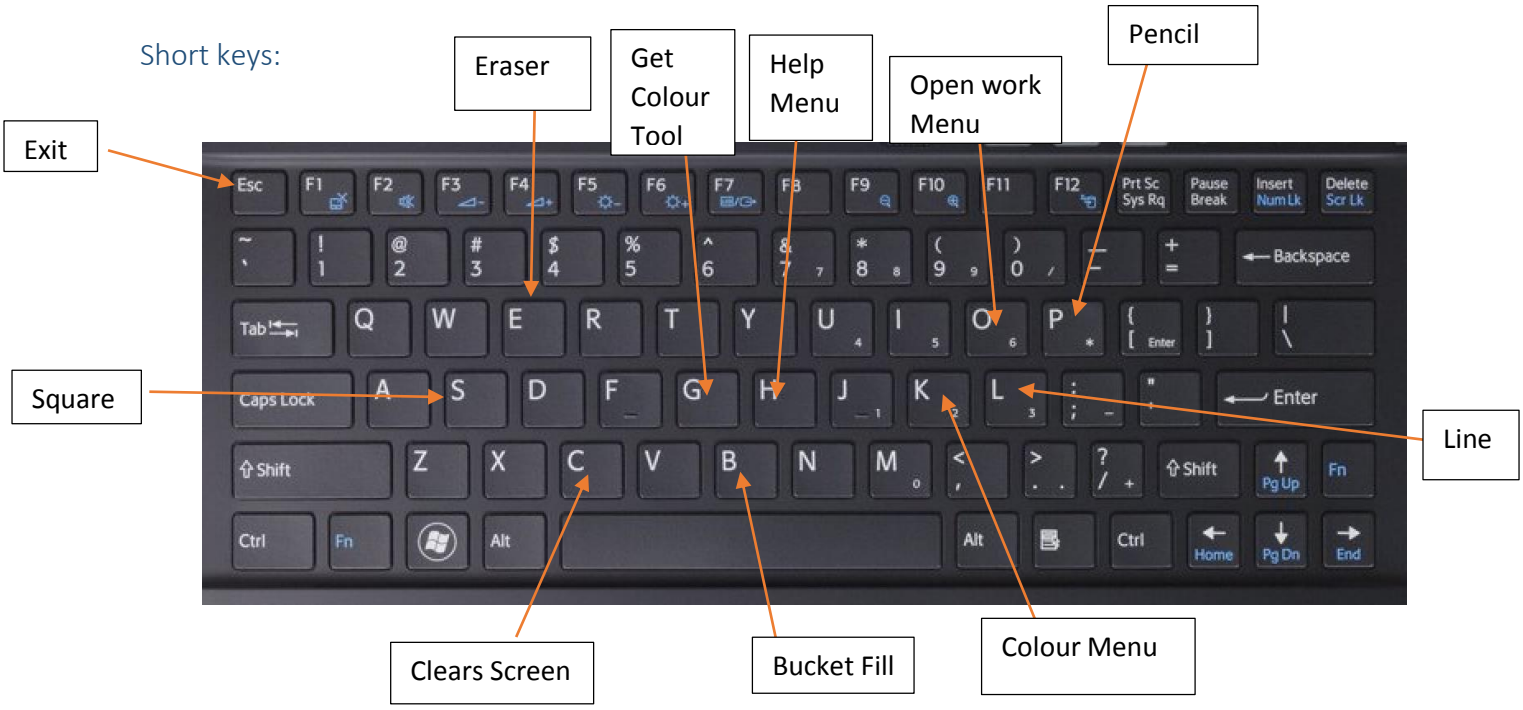
User Manual:

Screen Layout:

1) Pressing on the icons on the left will change your current tool into the icon that was pressed
2) Pressing on the coloured square on the bottom left will take you into a colour menu, where there is a large variety of colours for the user to choose from. As soon as the user chooses a colour, the user will be returned to his drawing.
3) Pressing on the colours on the top will change the users current colour to the colour pressed
4) The bottom toolbar has some additional options including saving and opening work, help menu and exit. Exiting will save your work automatically into lastsave so don't worry about saving.
5) When using Short Key for New Screen, double tap c just in case of accidental press
6) Enjoy and have fun

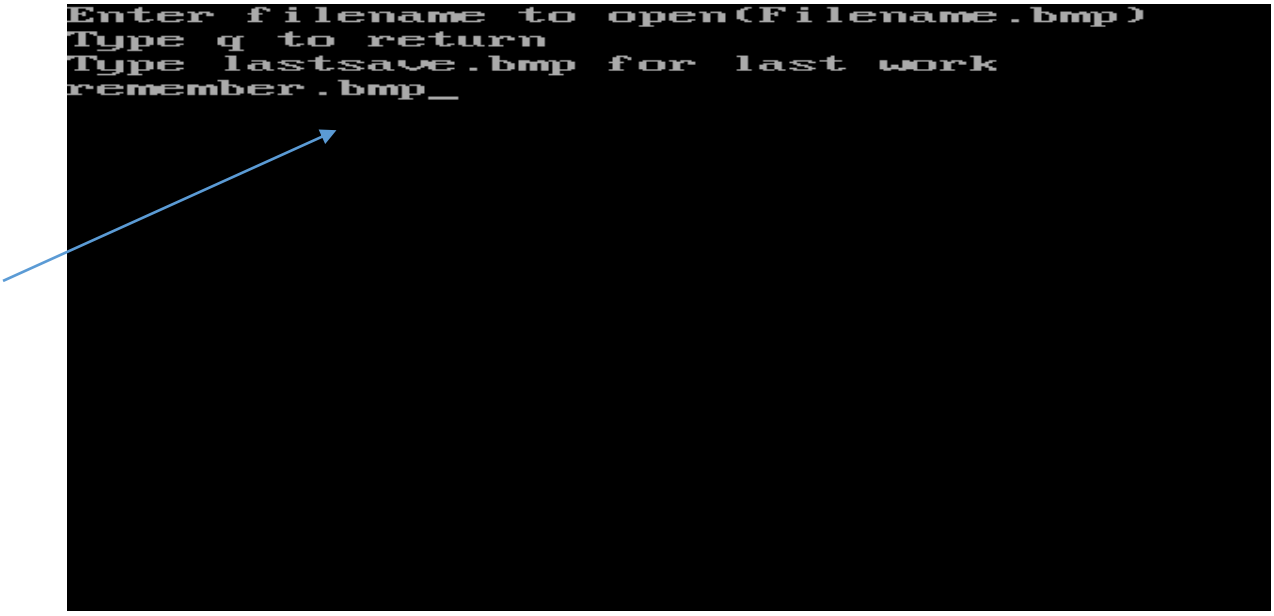


Short keys:

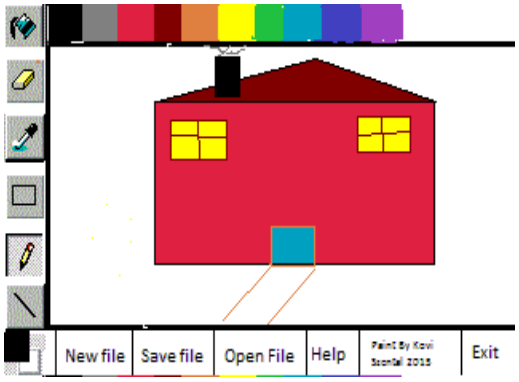


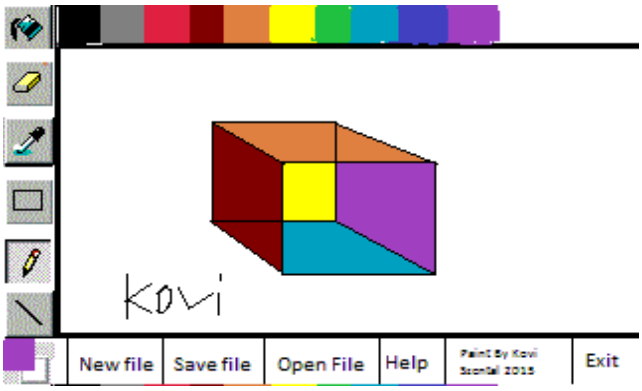
Saving and Opening Files:

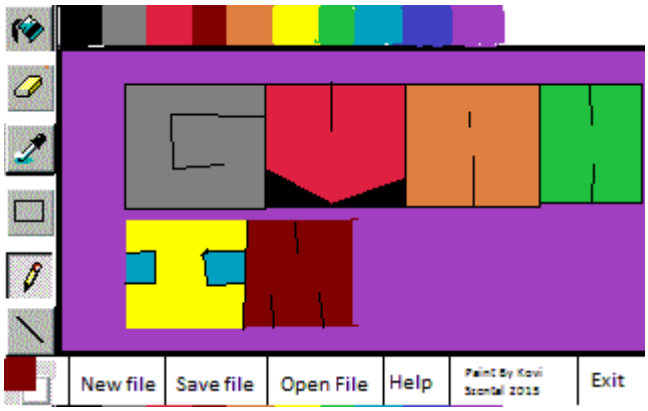
When saving and opening a file, remember to add .bmp at the end of the file.



Some Things you can draw:

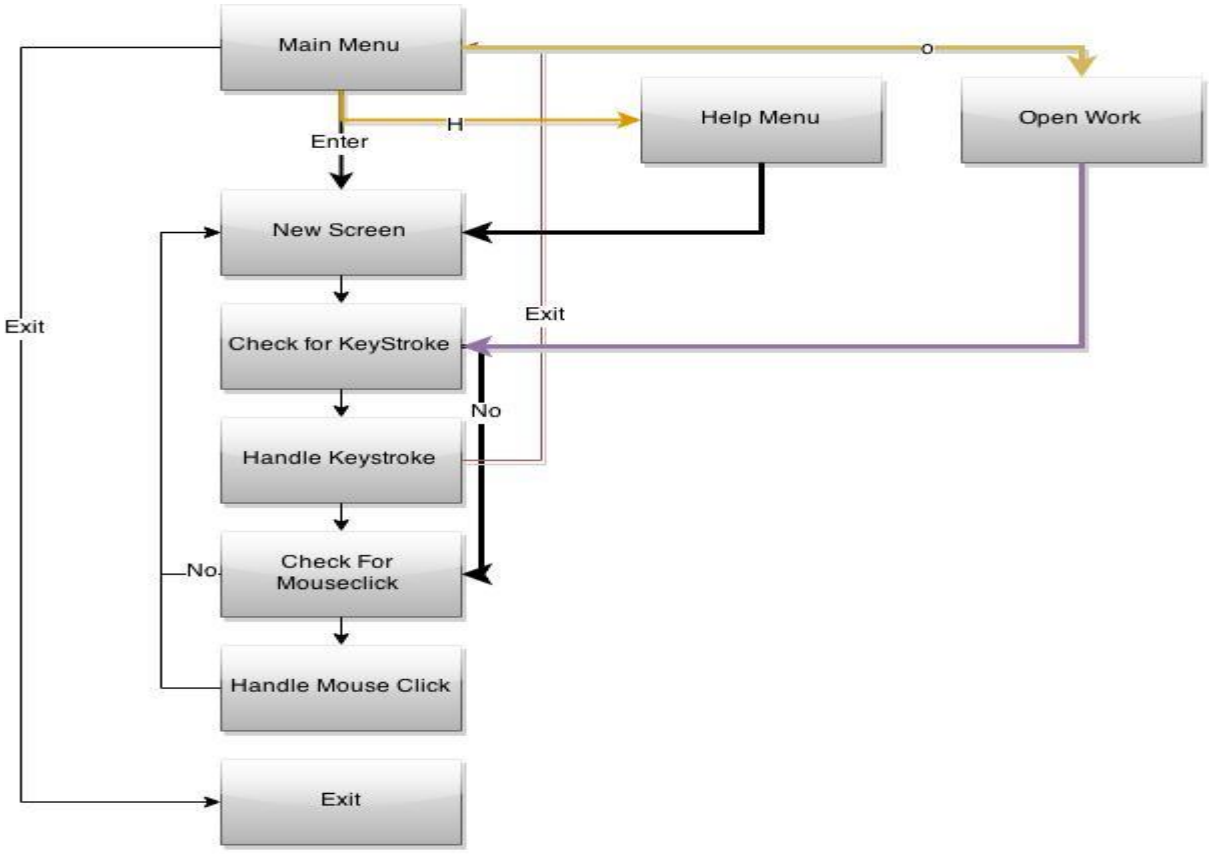




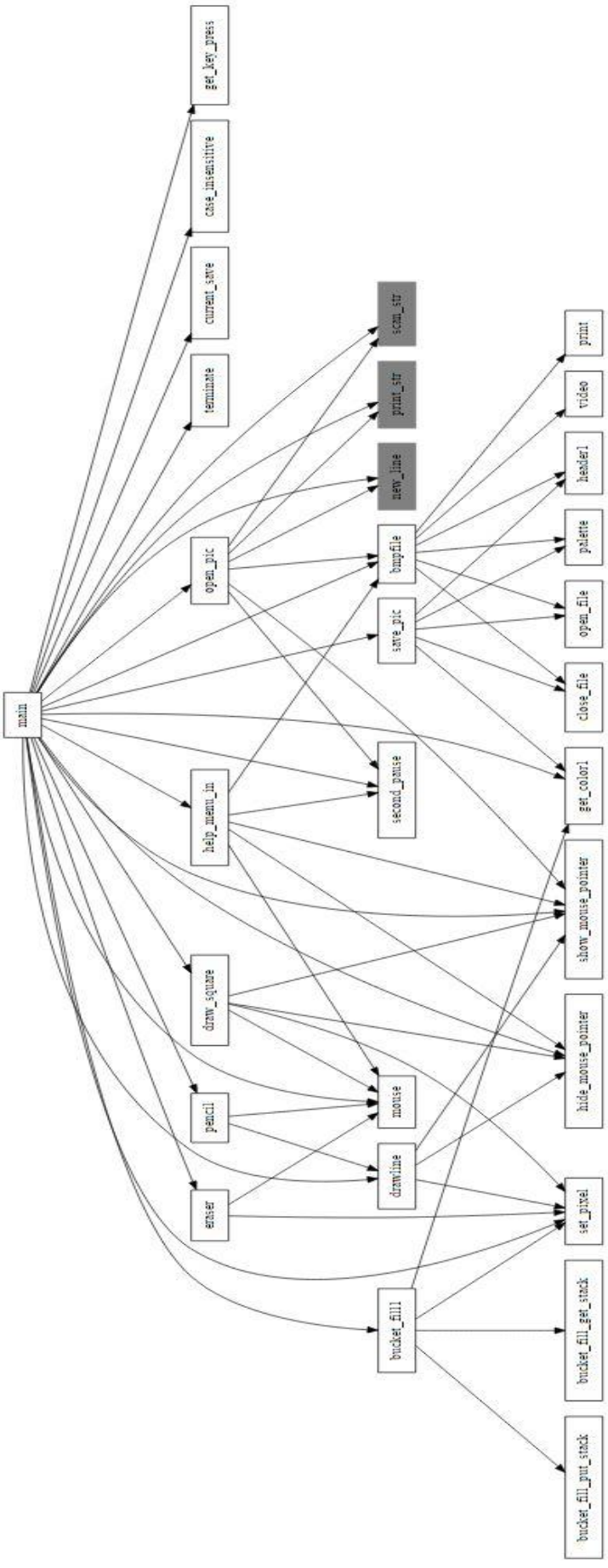


Program build:

Program Flow Chart:



Function Call Tree:



Programs Main Functions:

Set_Pixel:

The Function places a coloured pixel on the screen at a given coordinate with a given color.

Bmp_File:

The Function receives a name of a BMP file, opens the file and displays the picture on screen.

Get_Key_Press:

The Function checks for input from keyboard and returns key stroke if found.

Mouse:

The Function checks the mouse pointers status, returns its coordinates and if any button is Pressed down.

Help_Menu_In:

The Function opens a help menu.

Get_Color:

The Function checks the colour of pixel at a certain coordination

DrawLine:

The Function draws a line between two given points using Bressenham Line Algorithm(See later)

Pencil:

The Function draws a pixel on screen wherever the left mouse button is down

Eraser:

The Function draws white pixels on screen wherever the left mouse button is down

DrawSquare:

The Function draws a square between two given points

Bucket_Fill:

The Function fills the screen using the Bucket Fill Algorithm (See later on).

Save_Pic:

Creates a file and saves current picture to that file.

Open_Pic:

Opens Open menu.

Main Program Variables and File Strings:

Color:

Stores the current colour being used.

Mode:

Stores the current mode.

Point1x, Point2x,Point1y, Point2y:

Stores coordinates

FileHandle:

Stores the File Handle of a file

Intro.Bmp:

Starting Menu Picture

Help.Bmp, Help2.Bmp:

Help Menus Pictures

Draw.bmp:

Empty Paint screen

Colorp.Bmp:

Colour Menu

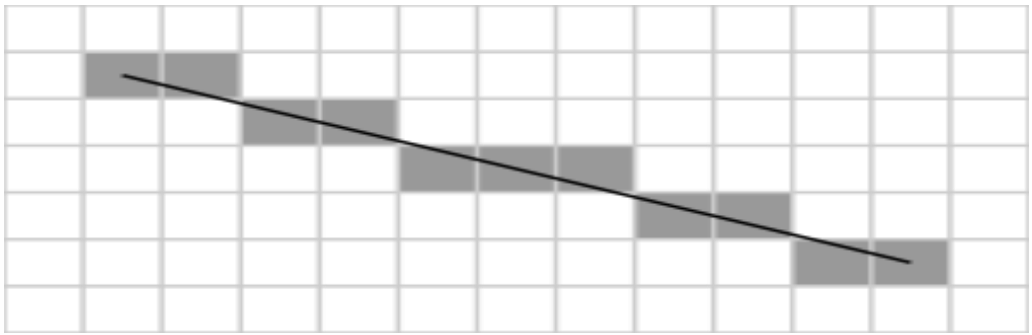
LastSave.bmp:

Last Saved work

Algorithms:

Bressenham Line Algorithm:

The Bressenham Line Algorithm is an algorithm for drawing a line between two points. The problem with using a normal line algorithm is that the line could be going between two pixels at once, which isn't possible because every pixel represents a point,not half a pixel, so either the next point in the line has to stay at the current y/x value(depending on steepness of line) or should be increased/decreased by 1. The algorithm is used to figure that out..



Bucket Fill Algorithm:

The Bucket Fill Algorithm is used for the bucket tool. The tool gets the colour of the pixel pressed on, and then changes all adjacent pixels that are the same colour to the new colour chosen. This works with recursion. Recursion is the calling of a function on itself. In the case of bucket fill, recursion is used to check each pixel and then every adjacent pixel of that pixel and keeps on going. If the colour of the pixel being checked doesn't match the one pressed on or is outside the border of the screen, the function will return to the last place it was called. So if I start at (1, 1), and then check (1,2) which is x+1 and that doesn't fill the criterion, the function will return to the place after it was called. So after x+1 was called, y+1 was called so now the point being checked will be (2, 2).

Problems and Solutions

Square, Get Colour , Eraser :

The reason there are three things under the same category is because they all had the same problem. With Get Colour, whenever the interrupt for retrieving pixel colour would be used with the mouse coordinates of a mouse click, the colour would always come back as black. Drawing a square also had a problem that wherever the mouse ended up for the second point, a gap was appearing at that same point after the square was drawn. The eraser had a similar problem where the bottom left corner kept getting wiped out. The problem in all cases was that the mouse pointer was being saved into the video memory so the mouse pointer was overwriting the pixels at the same position in the memory.

Solution:
Hide the Mouse Pointer, Draw Square/Get Colour/Erase, Show Mouse pointer

Bucket_Fill:

As mentioned above, bucket fill uses recursion to spread out across the screen. The problem was that every time a function is called, it pushes the IP into the stack and pops it once the function is returned. The problem with the recursion in bucket fill was that since the function pushed the IP several times into the stack without taking it back out, the amount of IPs being pushed into the stack was bigger than the stack itself, causing the stack to overflow.

Solution:
Made an array to be used instead of the stack itself. Instead of the function calling on itself, the function saves the place it needs to return to if function is supposed to return to and then jumps back to the beginning of the function. Because even then, it is still using too much space in an array, 4 different numbers are put into 1 byte(by using shift) instead of putting 1 number in 1 byte using a quarter of space than required before.

Saving Photo:

Solution:
Copied the Header and Palette of the original screen which has the exact same colour palette to the file and then copied the screen itself to the file.

Future Additions:

Circle:

Drawing a circle. This can be done with the Bressenham Circle Algorithm

Picture:

Importing a picture into the users work.

Line Thickness:

Changing the thickness of a line. This can be done by saving the current thickness of the line and then increment the x/y value(depending on angle of line) for each pixel on the line by that amount and print a pixel there, then that pixel-1 until it returns to the original pixel

Colour 2:

Adding an extra colour for the user. Color2 could be used as the colour for the inside of a shape. This can be done just by making another data variable with another colour inside. Inside the shape, “Colour” will be the colour of the border and “Colour 2” being the colour of the inside fill.

SprayPaint:

Addition of the spray paint tool. This can be done by adding another mode (7) which randomizes pixels in a certain radius from mouse click and paints the current colour on the randomized pixel.

Thank You:

Huge Thank You to Dov Feldstern, Yair Mirsky and Chaoul Charmoula for all the help and effort spent teaching and assisting me throughout the year with assembly . I really appreciate all the time spent during and after-hours to help and give me advice.

Thanks a lot!!!!!!!!!!!!!!!!!!!