Computer Graphics Lab Manual

Lab – 1: Installation and Basic Drawing using graphics.h

Objective

- To install and configure graphics.h library in C++.
- To understand **pixel plotting** using putpixel().
- To draw **lines and basic 2D shapes** (square, rectangle) using loops.

Software & Tools Required

- C++ Compiler (Turbo C++ / MinGW + Code::Blocks / Dev C++).
- Code Editor (Vs Code)
- graphics.h header file.
- winbgim.h & libbgi.a library.

Theory

1. **Pixel:**
    - A pixel is the smallest unit of a display screen.
    - In computer graphics, an image is formed by plotting pixels at specific (x, y) coordinates.

2. **Coordinates in graphics.h:**
    - Origin (0,0) is at **top-left corner**.
    - x increases → **right side**.
    - y increases → **downwards**.
3. **Important Functions:**
    - initgraph(&gd, &gm, "") → Initializes graphics mode.
    - putpixel(x, y, color) → Plots a pixel at (x, y) with given color.
    - line(x1, y1, x2, y2) → Draws a line.
    - rectangle(left, top, right, bottom) → Draws a rectangle.

Procedure

1. Install and configure graphics.h library.
2. Write a program to test using putpixel() at different coordinates.
3. Observe how x and y values affect pixel position.
4. Write a loop to draw a line using consecutive pixels.
5. Extend the logic to draw **square** and **rectangle** using multiple lines/loops.

Programs & Output

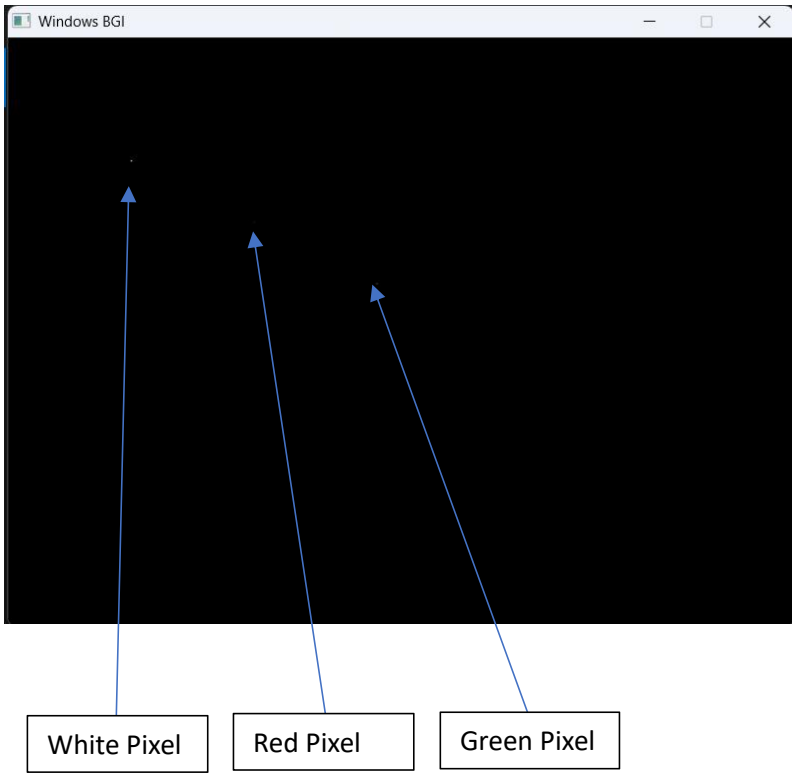
Program 1: Pixel Plotting

```
#include <graphics.h>
#include <conio.h>
int main() {
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "");

    putpixel(100, 100, WHITE);
    putpixel(200, 150, RED);
    putpixel(300, 200, GREEN);

    getch();
    closegraph();
    return 0;
}
```

Output: Different pixels plotted at given coordinates.



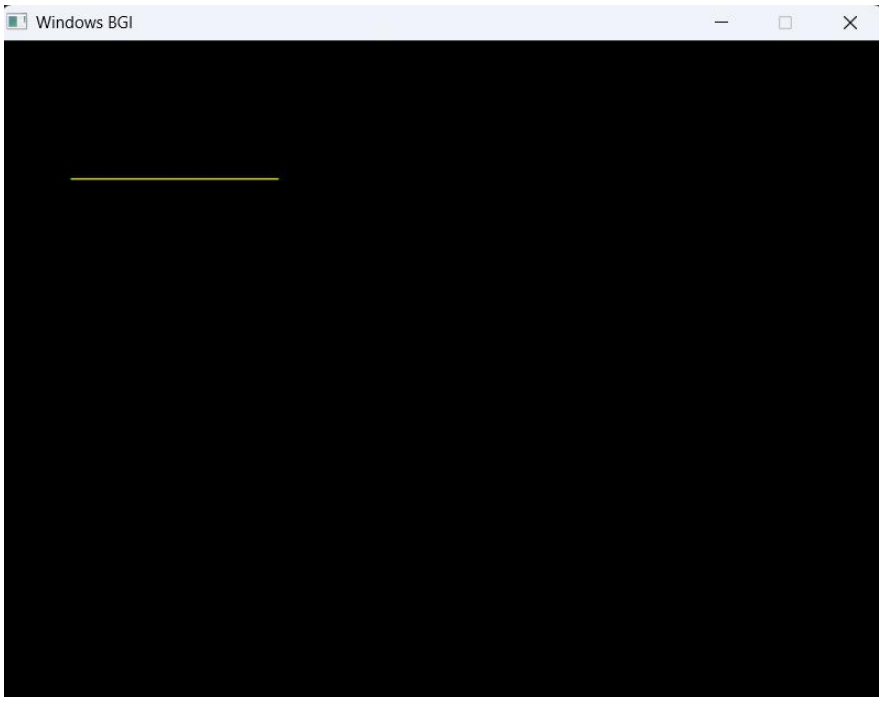
Program 2: Drawing a Line using putpixel()

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

int main() {
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "");

    for (int x = 50; x <= 200; x++) {
        putpixel(x, 100, YELLOW);
    }
    getch();
    closegraph();
    return 0;
}
```

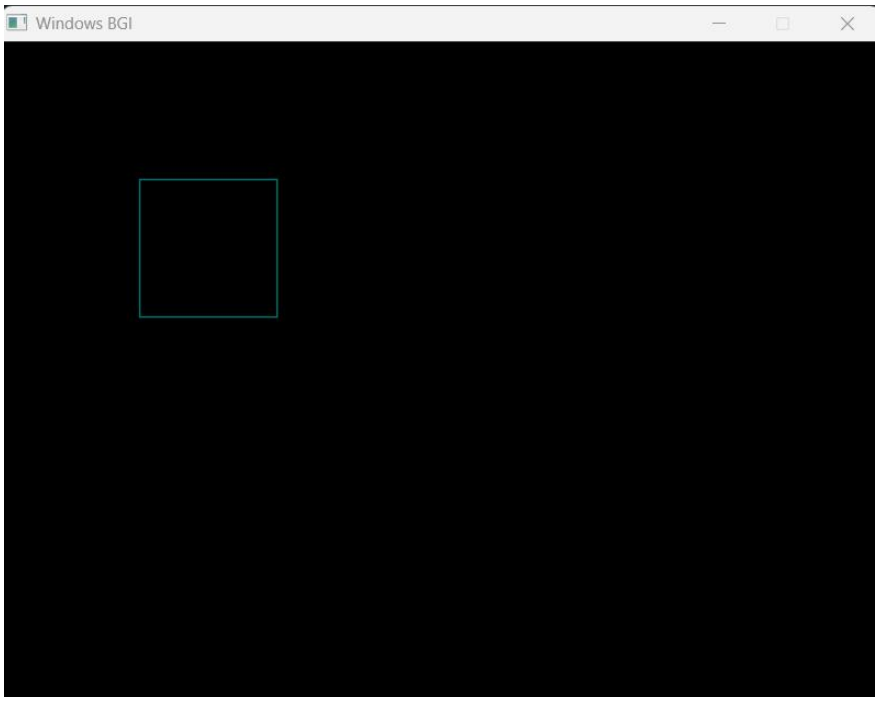
Output: A horizontal line from (50,100) to (200,100).



Program 3: Drawing a Square using Loops

```
#include <graphics.h>
#include <conio.h>
int main() {
    int gd = DETECT, gm;
    initgraph(&gd, &gm, "");
    // Top line
    for (int x = 100; x <= 200; x++) putpixel(x, 100, CYAN);
    // Bottom line
    for (int x = 100; x <= 200; x++) putpixel(x, 200, CYAN);
    // Left line
    for (int y = 100; y <= 200; y++) putpixel(100, y, CYAN);
    // Right line
    for (int y = 100; y <= 200; y++) putpixel(200, y, CYAN);
    getch();
    closegraph();
    return 0;
}
```

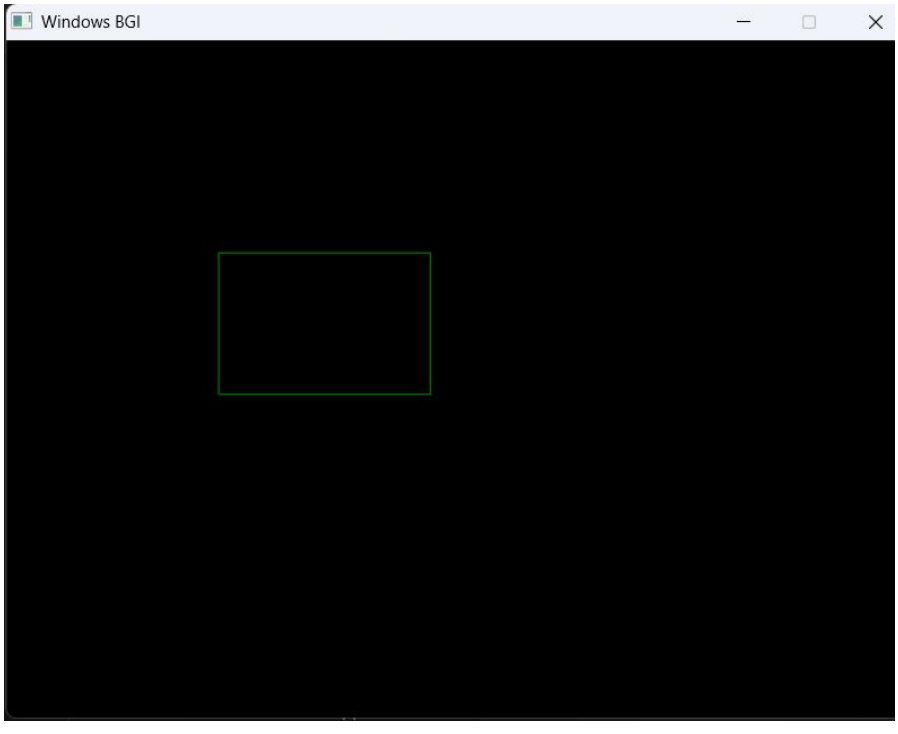
Output: A square of side length 100 pixels.



Program 4: Drawing Rectangle using Loops

```
#include <graphics.h>
#include <conio.h>
int main() {
   int gd = DETECT, gm;
   initgraph(&gd, &gm, "");
   // Top line
   for (int x = 150; x <= 300; x++) putpixel(x, 150, GREEN);
   // Bottom line
   for (int x = 150; x <= 300; x++) putpixel(x, 250, GREEN);
   // Left line
   for (int y = 150; y <= 250; y++) putpixel(150, y, GREEN);
   // Right line
   for (int y = 150; y <= 250; y++) putpixel(300, y, GREEN);
   getch();
   closegraph();
   return 0;
}
```

Output: A rectangle with width = 150 pixels and height = 100 pixels.



Observations

- Pixel coordinates (x, y) affect placement on the screen.
- x moves horizontally, y moves vertically.
- Shapes like **lines, squares, rectangles** can be constructed using loops of putpixel().

Conclusion

- Successfully installed and tested graphics.h.
- Understood pixel plotting and basic coordinate system.
- Implemented drawing of **line, square, and rectangle** using loops.
