2.1 Graphics Libraries

C++ graphics using **graphics.h** functions can be used to draw different shapes, display text in different fonts, change colors, and many more. Using functions of graphics.h in C++ compiler can make graphics programs, animations, projects, and games. You can draw circles, lines, rectangles, bars, and many other geometrical figures. You can change their colors using the available functions and fill them. Following is a list of functions of graphics.h header file. Every function is discussed with the arguments it needs, its description, possible errors while using that function and a sample C++ graphics program with its output.

Syntax: **#include<graphics.h>**

2.2 Initializing C++ Graphics Mode:

1.InitGraph ()

The computer display system must be initialized into graphics mode before calling the graphics function.

The "**initgraph**" function is used to initialize the display into graphics mode. This function is a part of the "**graphics.h**" header file. So this file is included in the program before executing "initgraph" function. The syntax of initgraph" function is:

Void initgraph(int &Graphriver, int &Graphmode,Char "pathtoDriver");

Graph Driver OR (gd):

Represents the graphics driver installed on the computer. It may be an integer variable or an integer constant identifier, e.g. CGA, EGA, SVGA, etc.

The graphics driver can also be automatically detected by using the keyword “DETECT”. Letting the compiler detect the graphic driver is known as auto-detect.

If the driver is to be automatically detected, the variable driver is declared as of integer type and **DETECT** value is assigned to it as shown below.

int driver, mode;

driver = DETECT;

This statement must be placed before “initgraph” function. When the above statement is executed. the computer automatically detects the graphic driver and the graphics mode.

Graph Mode OR (gm):

Represents output resolution on the computer screen. The normal mode for VGA is VGAHI. It gives the highest resolution If the driver is auto-detected, then its use is optional. The computer automatically detects the driver as well as the mode.

& :

represents the addresses of constant numerical identifiers of driver and mode. If constants (e.g., VGA, VGAHI) are used, then "&" operator is not used as shown below:

initgraph (VGA, VGAHI, "path");

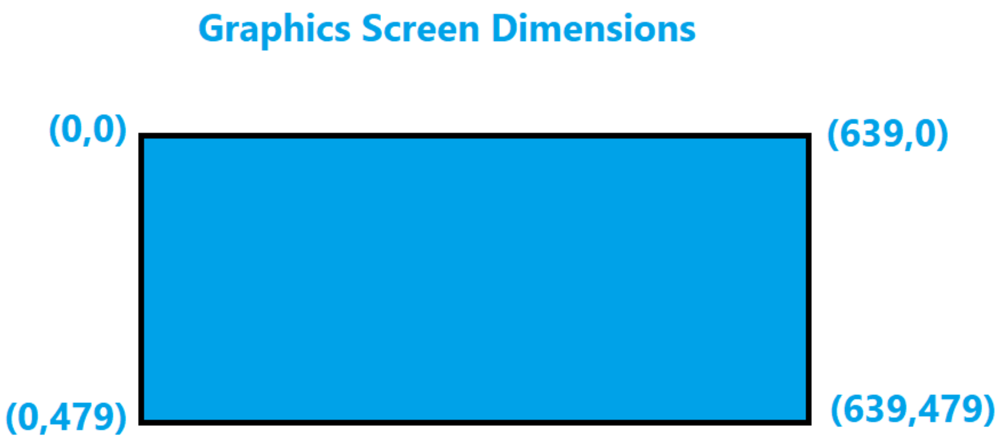
Path to driver:

Represents the path of graphic drivers. It is the directory where the BGI files are located. Suppose the BGI files are stored in "C:\TC\BGI", then the complete path is written as:

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

The use of double backslash "\" is to be noted. One backslash is used as an escape character and the other for the directory path. If the BGI files are in the current directory, then the path is written as:

Initgraph(&gd,&gm,"");



2.Grapherrormsg ():

This function used to return the error message.

the syntax: **Char *grapherrormsg(int errorcode);**

for example, if we want to print the error message write the following:

```
int gd, gm, errorcode;
initgraph(&gd, &gm, "C:\\TC\\BGI");
errorcode = graphresult();
if(errorcode != grOk)
 {
   printf("Graphics error: %s\n", grapherrormsg(errorcode));
   exit (1);

 }
```

3.Graphresult ():

This function is used to return the error code at any A drawing process when an error occurs. When no error occurs in the drawing operations, the function returns the value **(grok).** For example

```
int errorcode = graphresult();
if (errorcode != grOk)
{
  printf(grapherrormsg(errorcode);
  getch();
  return 0;
 }
```

4. Closegraph():

The **close graph** function is used to restore the screen to text mode. When graphics mode is initialized, memory is allocated to the graphics system. When "closegraph" function is executed, it de-allocates all memory allocated to the graphics system. This function has usually used at the end of the program.

Its syntax is **Void closegraph();**

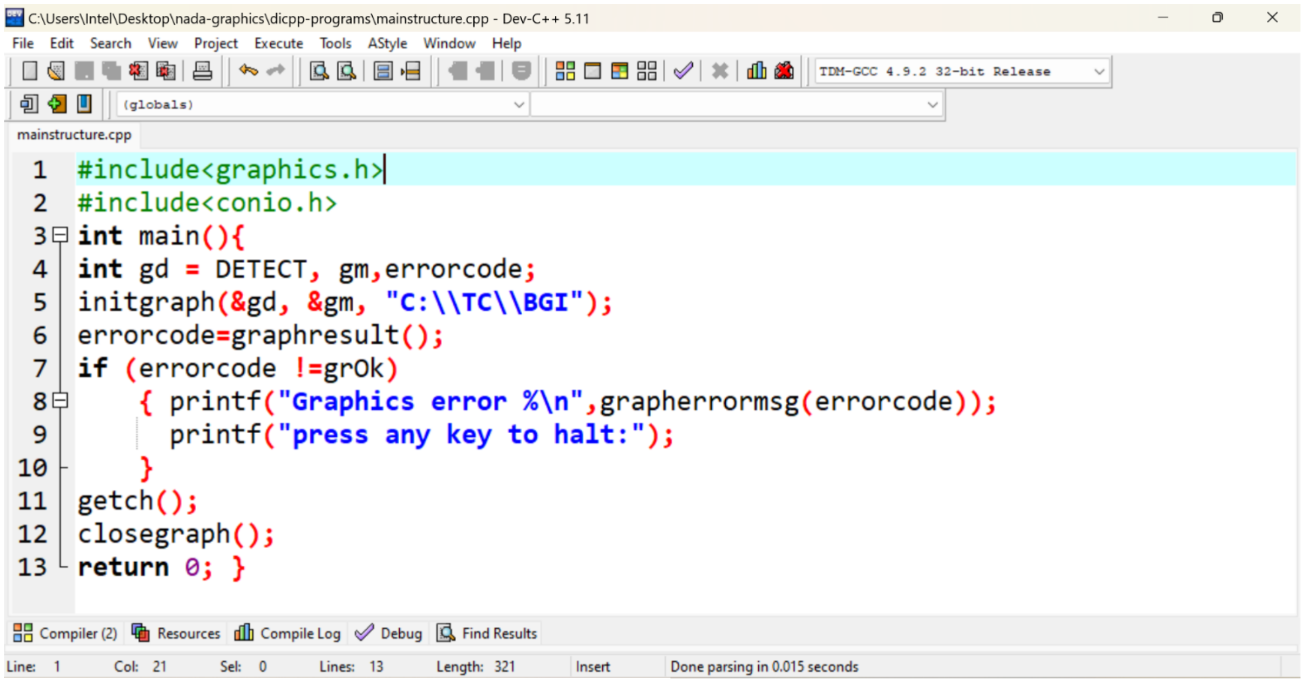
For example: if we want to convert the computer system from (**text mode**) to (**graphics mode**) and write the sentence **(Press any Key to Close the Graphics Mode...**), then close the graphics mode and return the system to the text mode.

```
initgraph(&gd, &gm, "");
outtext("Press any key to close the Graphics Mode...");
getch();
closegraph();
```

5.cleardevice ():

The "cleardevice" function is used to clear the screen in graphics mode. It is similar to "clrscr" function that is used to clear the screen text mode. Its syntax is: **cleardevice();**

The general structure of all graphic programs in C++:


C:\Users\Intel\Desktop\nada-graphics\dicpp-programs\mainstructure.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 32-bit Release
(globals)
mainstructure.cpp
Compiler (2) Resources Compile Log Debug Find Results

```
#include<graphics.h>
#include<conio.h>
int main(){
int gd = DETECT, gm,errorcode;
initgraph(&gd, &gm, "C:\\TC\\BGI");
errorcode=graphresult();
if (errorcode !=grOk)
     { printf("Graphics error %\n",grapherrormsg(errorcode));
       printf("press any key to halt:");
     }
getch();
closegraph();
return 0; }
```

```
Line: 1    Col: 21    Sel: 0    Lines: 13    Length: 321    Insert    Done parsing in 0.015 seconds
```

HOMEWORK

What is the output of this code?

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
void main ()
{
int gd=DETECT,gm;
initgraph(&gd,&gm, "");
outtext("Enter any key to come outside from the graphics mode");
getch();
closegraph();
printf("Text Mode ");
}
```