Vector Graphic with Polymorphic Inheritance

Submission Deadline: 23:59, April 15 2017 **Earnings:** 9% of your final grade
NOTE: Plan to finish a few days early to avoid last minute hardware/software holdups.

The code in this assignment must be your own work. It must not be code taken from another student or written for you by someone else, even if you give a reference to the person you got it from (attribution); if it is not entirely your own work it will be treated as plagiarism and given a fail mark, or less.

Purpose: This assignment is a direct continuation of assignment 2 that uses polymorphic inheritance. It stores a dynamic array of pointers of type `GraphicElement*` but now the objects being pointed to are actually of type `GeometricElement` or `TerrainElement` that are derived from the abstract base class `GraphicElement`. This dynamic array is of unlimited size and grows by one each time a new `GraphicElement` is added and shrinks by one each time a `GraphicElement` is deleted so there is no unused dynamic memory at any time.
Polymorphism ensures that when one of the `GraphicElement*` pointers calls its polymorphic function `IntensifyColour(),` as in the statement

```
Elements[i]->IntensifyColour();
```

the overridden version appropriate to the actual object type is called. Your code uses this statement.

Also, `name` is now an instance of the string class, which has overloaded operators that make handling strings easier. As in assignment 2, the vector template class is used to buffer Lines.
There are also two new classes: `coloref` and `texture` that are used in the derived classes `GeometricElement` and `TerrainElement. coloref` is used to represents rgb colours in simple geometry and `texture` represents an image with a background `coloref` that is mapped onto terrain.
The `IntensifyColour()` polymorphic function works differently for the two derived classes `GeometricElement` and `TerrainElement`. In `GeometricElement` the RGB values are replaced. In `TerrainElement` they are incremented and since they are unsigned char, the values go up to a max of 255 and then increment to 0.
To simplify the assignment and focus on the polymorphism, some operators and functions from assignment 2 have been removed.

Part of the code is shown on the next page; it is also on the Web Site in text files that you can copy and paste from so you don't make any mistakes. You must use this code without modification or additions because I will use it to mark your assignment. Your task is to implement the member functions that are declared in the Attributes.h, Line.h, GraphicElement.h, GeometricElement.h, TerrainElement.h and VectorGraphic.h header files and not add any new ones. The code you write and submit is in the files Attributes.cpp, Line.cpp, GraphicElement.cpp, GeometricElement.cpp, TerrainElement.cpp and VectorGraphic.cpp. There are no global variables, defines, constants or macros in your .cpp files, only the member function definitions and header file includes.

In this assignment, when the application is running the user can

- Add a new Graphic Element (together with its lines) to the Vector Graphic
- Delete a Graphic Element
- Print all the details of the Graphic Elements in the Vector Graphic
- Intensify the colours of a Graphic Element

An example of the output of the running application is given at the end. Yours must work identically and produce identical output.

Note the following:

- you must use C++ syntax including `new` and `delete` for memory allocation and `cin` and `cout` etc.
- there are no global variables, defines, constants or macros in your .cpp files, only the member function bodies and header file includes,

- When the application terminates it releases **all** dynamically allocated (or you lose 30%).

See the Marking Sheet for how you can lose marks, but you will lose at least 60% if: you change or add to the supplied code, it fails to build in Visual Studio 2013, it crashes in normal operation (such as printing from an empty list or adding elements etc.), it doesn't produce the example output.
Part of the code is shown on the next page. You MUST use this code **without modification.** Your task is to add the implementation of the functions that are declared using the style of the posted Submission Standard.

What to Submit : Use Blackboard to submit this assignment as a zip file (**not** RAR) containing only the source code files (Attributes.cpp, Line.cpp, GraphicElement.cpp, GeometricElement.cpp, TerrainElement.cpp and VectorGraphic.cpp). The name of the zipped folder **must** contain your name as a prefix so that I can identify it, for example using my name the file would be tyleraAss3CST8219.zip. It is also vital that you include the Cover Information (as specified in the Submission Standard) as a file header in your source file so the file can be identified as yours. Use comment lines in the file to include the header. Before you submit the code,

- check that it builds and executes in Visual Studio 2013 as you expect - if it doesn't build for me, for whatever reason, you get a deduction of at least 60%.
- make sure you have submitted the correct file – if I cannot build it because the file is wrong or missing from the zip, even if it's an honest mistake, you get 0.
- Due to Finals it can't be late. Don't send me file(s) as an email attachment – it will get 0.

Example code – I will use it to mark your assignment. Don't change or add to it.

```
// Attributes.h

#ifndef ATTRIBUTES
#define ATTRIBUTES

class coloref
{
        unsigned char red;
        unsigned char green;
        unsigned char blue;
public:
        coloref() :red(0), green(0), blue(0){}
        coloref(unsigned char red,unsigned char green,
unsigned char blue) :red(red), green(green), blue(blue){}
        coloref(coloref& c): red(c.red), green(c.green),
blue(c.blue){}
        coloref& operator=(coloref& RCR);
        void operator++();
        friend ostream& operator<<(ostream&, coloref&);
};

class texture
{
        string textureFileName;
        coloref bkgColour;
public:
        texture(){}
        texture(string s, coloref c =
coloref()):textureFileName(s),bkgColour(c){}
        texture(texture&
t):textureFileName(t.textureFileName),bkgColour(t.bkgColour){}
        void IntensifyColour(int);
        friend ostream& operator<<(ostream&, texture&);
};

class Point
{
        int x, y;
public:
        Point() :x(0), y(0){}
        Point(int x, int y) :x(x), y(y){}
        friend ostream& operator<<(ostream&, Point&);
};

#endif
```

```
// Line.h
#ifndef LINE
#define LINE

class Line
{
        Point start;
        Point end;
public:
        Line() :start(), end(){}
        Line(Point start, Point end) :start(start),
end(end){}
        friend ostream& operator<<(ostream&, Line&);

};

#end
```

```
// GraphicElement.h

#ifndef GRAPHICELEMENT
#define GRAPHICELEMENT

class GraphicElement
{
        vector<Line> Lines; // a vectorof Lines
protected:
```

```
// GeometricElement.h


#ifndef GEOMETRICELEMENT
#define GEOMETRICELEMENT

class GeometricElement : public GraphicElement
{
        coloref colour;
```

```
          string name;
public:
          GraphicElement(){};
          GraphicElement(string s) :name(s){}
          GraphicElement(vector<Line>, string);
          GraphicElement(GraphicElement&);
          virtual~GraphicElement(){}
          friend ostream& operator<<(ostream&,
GraphicElement&);
          virtual void IntensifyColour() = 0;
};

#endif
```

```
public:
          GeometricElement():GraphicElement(){}
          GeometricElement(string s) :GraphicElement(s){}
          GeometricElement(coloref c,vector<Line> v,string
s):GraphicElement(v,s),colour(c) {}
          GeometricElement(GeometricElement& RGE)
:GraphicElement(RGE), colour(RGE.colour){}
          friend ostream& operator<<(ostream&,
GeometricElement&);
          void IntensifyColour();
};


#endif
```

```
// TerrainElement.h

#ifndef TERRAINELEMENT
#define TERRAINELEMENT


class TerrainElement: public GraphicElement
{
          texture terrain;
public:
          TerrainElement() :GraphicElement(){}
          TerrainElement(string s) :GraphicElement(s){}
          TerrainElement(texture t, vector<Line> v, string s)
:GraphicElement(v,s), terrain(t){}
          TerrainElement(TerrainElement& t)
:GraphicElement(t),terrain(t.terrain){}
          friend ostream& operator<<(ostream&,
TerrainElement&);
          void IntensifyColour();
};
#endif
```

```
// VectorGraphic.h

#ifndef VECTORGRAPHIC
#define VECTORGRAPHIC

class VectorGraphic
{
          GraphicElement** Elements;

public:
          unsigned int numElements;
          VectorGraphic():numElements(0),Elements(nullptr){}
          ~VectorGraphic()
          {
                    for (int i = 0; i < numElements; i++)
                              delete Elements[i];
                    delete[] Elements;
          }
          void AddGraphicElement();
          void DeleteGraphicElement();
          void IntensifyColour();

          friend ostream& operator<<(ostream&,
VectorGraphic&);
};

#endif
```

```
// ass3 W17

#define _CRT_SECURE_NO_WARNINGS
#define _CRTDBG_MAP_ALLOC     // need this to get the line identification
//_CrtSetDbgFlag(_CRTDBG_ALLOC_MEM_DF|_CRTDBG_LEAK_CHECK_DF); // in main, after local declarations
//NB must be in debug build
#include <crtdbg.h>
#include <iostream>
#include <string>
#include <vector>
using namespace std;

#include "attributes.h"
#include "Line.h"
#include "GraphicElement.h"
#include "GeometricElement.h"
#include "TerrainElement.h"
#include "VectorGraphic.h"

          enum{ RUNNING = 1 };

          VectorGraphic Image;

          int main()
          {
                    char response;
                    _CrtSetDbgFlag(_CRTDBG_ALLOC_MEM_DF | _CRTDBG_LEAK_CHECK_DF);

                    while (RUNNING)
                    {
                              cout<<endl<<"Please select an option:\n"<<endl;
                              cout<<"1. Add a Graphic Element\n";
                              cout<<"2. Delete a Graphic Element\n";
                              cout<<"3. List the Graphic Elements\n";
                              cout<<"4. Intensify Colours\n";
                              cout<<"q. Quit\n";
                              cout<<"\nCHOICE: ";
                              cin>>response;

                              switch (response)
```

```
                        {
                        case '1':Image.AddGraphicElement(); break;
                        case '2':Image.DeleteGraphicElement(); break;
                        case '3':cout<<Image; break;
                        case '4':Image.IntensifyColour(); break;
                        case 'q':return 0;
                        default:cout<<"Please enter a valid option\n";
                        }
                        cout<<endl;
                }
                return 0;
        }
```

Example Output:

```
Please select an option:

1. Add a Graphic Element
2. Delete a Graphic Element
3. List the Graphic Elements
4. Intensify Colours
q. Quit

CHOICE: 1
Please enter the name of the new GraphicElement(<256 characters): left eye
How many lines are there in the new GraphicElement? 1
Please enter the x coord of the start point of line index 0 1
Please enter the y coord of the start point of line index 0 2
Please enter the x coord of the end point of line index 0 3
Please enter the y coord of the end point of line index 0 4
What is the type of the new element:  Geometric = 1, Terrain = 2?
1
please enter the rgb values of the coloref (255 max)
123
234
321


Please select an option:

1. Add a Graphic Element
2. Delete a Graphic Element
3. List the Graphic Elements
4. Intensify Colours
q. Quit

CHOICE: 1
Please enter the name of the new GraphicElement(<256 characters): nose
How many lines are there in the new GraphicElement? 1
Please enter the x coord of the start point of line index 0 1
Please enter the y coord of the start point of line index 0 2
Please enter the x coord of the end point of line index 0 3
Please enter the y coord of the end point of line index 0 4
What is the type of the new element:  Geometric = 1, Terrain = 2?
1
please enter the rgb values of the coloref (255 max)
111
222
333


Please select an option:

1. Add a Graphic Element
2. Delete a Graphic Element
3. List the Graphic Elements
4. Intensify Colours
q. Quit

CHOICE: 1
Please enter the name of the new GraphicElement(<256 characters): long hair
How many lines are there in the new GraphicElement? 4
```

```
Please enter the x coord of the start point of line index 0 1
Please enter the y coord of the start point of line index 0 2
Please enter the x coord of the end point of line index 0 3
Please enter the y coord of the end point of line index 0 4
Please enter the x coord of the start point of line index 1 5
Please enter the y coord of the start point of line index 1 6
Please enter the x coord of the end point of line index 1 7
Please enter the y coord of the end point of line index 1 8
Please enter the x coord of the start point of line index 2 9
Please enter the y coord of the start point of line index 2 0
Please enter the x coord of the end point of line index 2 1
Please enter the y coord of the end point of line index 2 2
Please enter the x coord of the start point of line index 3 3
Please enter the y coord of the start point of line index 3 4
Please enter the x coord of the end point of line index 3 5
Please enter the y coord of the end point of line index 3 6
What is the type of the new element:  Geometric = 1, Terrain = 2?
2
please enter the rgb values of the coloref (255 max)
222
222
234
please enter the name of the texture file
hair.txt


Please select an option:

1. Add a Graphic Element
2. Delete a Graphic Element
3. List the Graphic Elements
4. Intensify Colours
q. Quit

CHOICE: 3
VectorGraphic Report

Reporting Graphic Element #0
GEOMETRIC ELEMENT
rgb = 123,234,65
name = left eye
Element[0] : start is x = 1, y = 2. end is x = 3, y = 4


Reporting Graphic Element #1
GEOMETRIC ELEMENT
rgb = 111,222,77
name = nose
Element[0] : start is x = 1, y = 2. end is x = 3, y = 4


Reporting Graphic Element #2
TERRAIN ELEMENT
texture file name = hair.txt, background colour = rgb = 222,222,234
name = long hair
Element[0] : start is x = 1, y = 2. end is x = 3, y = 4
Element[1] : start is x = 5, y = 6. end is x = 7, y = 8
Element[2] : start is x = 9, y = 0. end is x = 1, y = 2
Element[3] : start is x = 3, y = 4. end is x = 5, y = 6



Please select an option:

1. Add a Graphic Element
2. Delete a Graphic Element
3. List the Graphic Elements
4. Intensify Colours
q. Quit

CHOICE: 4
```

```
Intensify Colours

Element #0
Geometric Element - Intensify Colour by Replacement
Name = left eye
Please state give the new rgb color values (between 0 and 255):
red = 222

green = 222

blue = 222

Element #1
Geometric Element - Intensify Colour by Replacement
Name = nose
Please state give the new rgb color values (between 0 and 255):
red = 0

green = 0

blue = 0

Element #2
Terrain Element - Intensify Colour by Increment
name = long hair
Please state give the increment to the rgb color values (between 1 and 255): 50


Please select an option:

1. Add a Graphic Element
2. Delete a Graphic Element
3. List the Graphic Elements
4. Intensify Colours
q. Quit

CHOICE: 3
VectorGraphic Report

Reporting Graphic Element #0
GEOMETRIC ELEMENT
rgb = 222,222,222
name = left eye
Element[0] : start is x = 1, y = 2. end is x = 3, y = 4


Reporting Graphic Element #1
GEOMETRIC ELEMENT
rgb = 0,0,0
name = nose
Element[0] : start is x = 1, y = 2. end is x = 3, y = 4


Reporting Graphic Element #2
TERRAIN ELEMENT
texture file name = hair.txt, background colour = rgb = 16,16,28
name = long hair
Element[0] : start is x = 1, y = 2. end is x = 3, y = 4
Element[1] : start is x = 5, y = 6. end is x = 7, y = 8
Element[2] : start is x = 9, y = 0. end is x = 1, y = 2
Element[3] : start is x = 3, y = 4. end is x = 5, y = 6



Please select an option:

1. Add a Graphic Element
2. Delete a Graphic Element
3. List the Graphic Elements
4. Intensify Colours
q. Quit
```

```
CHOICE:
```