24-780 Engineering Computation Problem Set 09

You need to create a ZIP file (It may appear as a compressed folder in Windows) and submit the ZIP file via the 24-780 Canvas. The file name of the ZIP file must be:

PS09-YourAndrewID.zip

For example, if your Andrew account is *hummingbird@andrew.cmu.edu*, the file name must be:

PS09-hummingbird.zip

If your ZIP file does not comply with this naming rule, you will automatically lose 5% credit from this assignment. If we are not able to identify who submitted the file, you will lose another 5% credit. If we finally are not able to connect you and the submitted ZIP file, you will receive 0 point for this assignment. Therefore, please make sure you strictly adhere to this naming rule before submitting a file.

The ZIP file needs to be submitted to the 24-780 Canvas. If you find a mistake in the previous submission, you can re-submit the ZIP file with no penalty as long as it is before the submission deadline.

Notice that the grade will be given to the final submission only. If you submit multiple files, the earlier version will be discarded. Therefore, if you re-submit a ZIP file, the ZIP file MUST include all the required files. Also, if your final version is submitted after the submission deadline, late-submission policy will be applied no matter how early your earlier version was submitted.

Make sure you upload your Zip file to the correct location. If you did not upload your assignment to the correct location, you will lose 5%.

The ZIP file needs to include:

ps9.cpp

Submission Due: Please see Canvas

PS9 Char-Bitmap class

The following C++ program includes a class called CharBitmap, which is to be completed. CharBitmap stores a bitmap defined by an array of char. Character code '1' means a black pixel, and other character codes mean an empty pixel. The class has following member functions in addition to copy constructor and copy-assignment operator. You can add your own functions.

CharBitmap	Constructor
~CharBitmap	Destructor
CleanUp	This function cleans up the bitmap. Memory chunk allocated for storing the bitmap must be deleted (returned to the operating system.)
GetWidth	Returns the width
GetHeight	Returns the height
SetBitmap	Makes a bitmap of bmpWid x bmpHei and copies pixels from bmp.
SetPixel	Set pixel value to a pixel at (x,y).
Print	Prints the contents to the console window.
Draw	Draws the bitmap to the graphics window. One pixel in this class must be expanded to 16x16 pixels on the graphics window.
GetPixel	Returns the character code at (x,y).

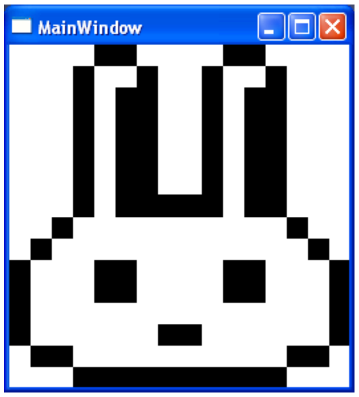
(a) Fill the functions that are not written yet.
(b) Assume that this class will be used by other team members. Add necessary modification to the code so that CharBitmap class is well protected. Const-qualifiers must be used wherever appropriate, and copy constructor and copy-assignment operator must be defined.
(c) SetPixel function must not crash if the given coordinate (x,y) are out of the bitmap stored in this object.

You can download the base code from the Canvas so that you don't have to type everything by yourself.

You must save the C++ program as ps9.cpp and must be submitted in the Zip file.

Test your code on the compiler server before submitting!

```
....11....11....
...1..1..1..1...
...1.11..1.11...
...1.11..1.11...
...1.11..1.11...
...1.11..1.11...
...1.11..1.11...
...1.11111.11...
..1..........1..
.1............1.
1...11....11...1
1...11....11...1
1..............1
1......11......1
.11..........11.
...1111111111...
```
(Console Output)



(Graphics window output)

(Base code)

```
#include <stdio.h>
#include "fssimplewindow.h"

class CharBitmap
{
protected:
    int wid,hei;
    char *dat;
public:
    CharBitmap();
    ~CharBitmap();
    void CleanUp(void);

    int GetWidth(void);
    int GetHeight(void);
    void SetBitmap(int bmpWid,int bmpHei,char bmp[]);
    void SetPixel(int x,int y,char c);

    void Print(void);
    void Draw(void);
    char GetPixel(int x,int y);
};

CharBitmap::CharBitmap()
{
    // Write this function.
}
CharBitmap::~CharBitmap()
{
    // Write this function.
}
void CharBitmap::CleanUp(void)
{
    // Write this function.
}
int CharBitmap::GetWidth(void)
{
    return wid;
}
int CharBitmap::GetHeight(void)
{
    return hei;
}
void CharBitmap::SetBitmap(int bmpWid,int bmpHei,char bmp[])
{
    // Write this function.
}
void CharBitmap::SetPixel(int x,int y,char c)
{
    // Write this function.
}
void CharBitmap::Print(void)
{
    int x,y;
    for(y=0; y<hei; y++)
    {
        for(x=0; x<wid; x++)
        {
            printf("%c",GetPixel(x,y));
        }
        printf("\n");
    }
}



void CharBitmap::Draw(void)
{
    // Write this function.
}

char CharBitmap::GetPixel(int x,int y)
{
    if(0<=x && x<wid && 0<=y && y<hei)
    {
        return dat[y*wid+x];
    }
    return 0;
}

int main(void)
{
    char pattern[]=
    {
        "....11....11...."
        "...1..1..1..1..."
        "...1.11..1.11..."
        "...1.11..1.11..."
        "...1.11..1.11..."
        "...1.11..1.11..."
        "...1.11..1.11..."
        "...1.11111.11..."
        "..1..........1.."
        ".1............1."
        "1..............1"
        "1..............1"
        "1..............1"
        "1......11......1"
        ".11..........11."
        "...1111111111..."
    };

    CharBitmap bmp;
    bmp.SetBitmap(16,16,pattern);
    bmp.SetPixel(4,10,'1');
    bmp.SetPixel(5,10,'1');
    bmp.SetPixel(4,11,'1');
    bmp.SetPixel(5,11,'1');
    bmp.SetPixel(10,10,'1');
    bmp.SetPixel(11,10,'1');
    bmp.SetPixel(10,11,'1');
    bmp.SetPixel(11,11,'1');
    bmp.Print();

    FsOpenWindow(16,16,256,256,1);
    while(FSKEY_NULL==FsInkey())
    {
        FsPollDevice();

        glClear(GL_DEPTH_BUFFER_BIT|GL_COLOR_BUFFER_BIT);
        bmp.Draw();
        FsSwapBuffers();

        FsSleep(10);
    }

    return 0;
}
```