Animation Project in C++

Due Time: 23.59, Sat 24 Feb 2018 **Earnings:** 8% of your final grade

NOTE: Plan to finish a few days early to avoid last minute hardware/software holdups for which no allowance is given.
NOTE: 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 is an adaptation of assignment #0 that uses the C++ language instead of C. You are to write the code in Visual Studio 2015 for a simple C++ language console application that holds the data of an animation application (there is no actual animation graphics in the assignment) using an array of pointers in dynamic memory. In the lab you will be shown what the application looks like when it's running.
Part of the code is shown on the next page; it is also on Blackboard in a text file that you can copy and paste. You **MUST** use this code **without modification (not a single character changed): no code added or removed, no new global variables, no macros, no defines and no statics**. Your task is to implement, using C++, the member functions of the Animation and Frame classes and not add any new ones. All your code is in the source code files Animation.cpp and Frame.cpp.

The Animation is a series of Frame objects held in dynamic (heap) memory with the pointers to the objects held in an array of pointers, also in dynamic memory. When the list runs it displays the details of each Frame at intervals of 1 second using the system clock (you are given the code for this). The heap memory occupied by the Frame pointers inflates itself when it is depleted. At the start before any Frames are added there is no dynamically allocated memory. When the first frame is added the dynamic memory for the Frame* array inflates by an amount specified by the BLOCK_SIZE parameter (BLOCK_SIZE is the additional number of Frame pointers). When this memory is used up it automatically inflates again and so on. See the example output for how it works.

You can:
- Add a new Frame to the Animation unless the frameName already exists
- Edit an existing Frame to change its frameName
- Delete all the Frames in the Animation leaving no dynamically allocated memory
- Run the Animation to show the list of Frame details one after another at 1 second intervals (use the code supplied in assignment 0)
- Quit

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:
- the files you submit must be named Frame.cpp and Animation.cpp,
- dynamic memory management is done with new and delete,
- all input and output is done with cin and cout
- you can only use functions like strlen() and strcpy() or similar etc. from the standard C library to handle strings of null terminated chars,
- When the application terminates it releases **all** dynamically allocated memory so it does not have a resource leak (or you lose 30%).

See the Marking Sheet for how you can lose marks, but you will lose marks if:
1. [-60%] you change the supplied code in any way at all (not a single character) - no code added or removed, no macros, no defines, no statics and no additional classes or global functions or variables,
2. [-60%] it fails to build in Visual Studio 2015,
3. [-60%] It crashes in normal operation (such as running an empty application etc.),
4. 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, in the Animation.cpp and Frame.cpp files, the implementation of the class member 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, not 9zip, not 7 zip) containing only the source code files Animation.cpp and Frame.cpp. The name of the zipped folder **must** contain your name as a prefix so that I can identify it, for example for CST8219 using my name the file would be tyleraAss1CST8219.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 2015 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 – no compromises.**

There is a late penalty of 25% per day. Don't send me the file as an email attachment – it will get 0.

Example code (also in a text file you can copy and paste on the web site). Don't change it.

```
#pragma once

class Animation {
	static const int BLOCK_SIZE = 64;
	char* animationName;
	unsigned int numFrames;
	Frame** frames;
public:
	Animation(char*);
	~Animation();
	void InsertFrame();
	void EditFrame();
	void DeleteFrames();
	void RunFrames();
};
```

```
#pragma once

class Frame {
	char* frameName;
public:
	Frame(char*);
	~Frame();
	bool FindFileName(char*);
	void ReplaceFrameName(char*);
	char* Report();
};
```

```
// ass1.cpp
#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
using namespace std;
#include <crtdbg.h>
#include <time.h>
#include <string.h>
#include <iostream>
#include "Frame.h"
#include "Animation.h"

int main(void)
{
	char response;
	bool RUNNING = true;
	Animation A("Animation#1");
	_CrtSetDbgFlag(_CRTDBG_ALLOC_MEM_DF | _CRTDBG_LEAK_CHECK_DF);

	while (RUNNING)
	{
		cout<<"MENU\n 1. Insert a Frame\n 2. Edit a Frame\n 3. Delete all the Frames\n 4.
Run the Animation\n 5. Quit\n";
		cin>> response;
		switch (response)
		{
		case '1':A.InsertFrame(); break;
		case '2':A.EditFrame(); break;
		case '3':A.DeleteFrames(); break;
```

```
                case '4':A.RunFrames(); break;
                case '5':RUNNING = false; break;
                default:cout<<"Please enter a valid option"<<endl;
                }
        }
        return 0;
```

Example Output:

```
MENU
 1. Insert a Frame
 2. Edit a Frame
 3. Delete all the Frames
 4. Run the Animation
 5. Quit
1
Insert a Frame in the Animation
Please enter the Frame name: Frame1
Number of Blocks = 1
MENU
 1. Insert a Frame
 2. Edit a Frame
 3. Delete all the Frames
 4. Run the Animation
 5. Quit
1
Insert a Frame in the Animation
Please enter the Frame name: Frame2
Number of Blocks = 1
MENU
 1. Insert a Frame
 2. Edit a Frame
 3. Delete all the Frames
 4. Run the Animation
 5. Quit
1
Insert a Frame in the Animation
Please enter the Frame name: Frame1
Frame name already exists
MENU
 1. Insert a Frame
 2. Edit a Frame
 3. Delete all the Frames
 4. Run the Animation
 5. Quit
2
Please enter the name of the Frame to edit: Frame3
Frame not found
MENU
 1. Insert a Frame
 2. Edit a Frame
 3. Delete all the Frames
 4. Run the Animation
 5. Quit
2
Please enter the name of the Frame to edit: Frame2
Found at index 1
What is the replacement name? Frame02
MENU
 1. Insert a Frame
 2. Edit a Frame
 3. Delete all the Frames
 4. Run the Animation
 5. Quit
4
Run the Animation
Animation name is Animation#1
Frame #0; time = 1;
Frame file name = Frame1;
Frame #1; time = 2;
Frame file name = Frame02;
```

```
MENU
 1. Insert a Frame
 2. Edit a Frame
 3. Delete all the Frames
 4. Run the Animation
 5. Quit
3
Delete all the Frames from the Animation
MENU
 1. Insert a Frame
 2. Edit a Frame
 3. Delete all the Frames
 4. Run the Animation
 5. Quit
4
No frames in the animation
MENU
 1. Insert a Frame
 2. Edit a Frame
 3. Delete all the Frames
 4. Run the Animation
 5. Quit
```