Animation in C++ with Inheritance

Due Time: 23.59, Sat 30 November 2019 - **Because of Finals this cannot be late**
Earnings: 9% of your final grade

NOTE: Plan to finish a few days early to avoid last minute hardware/software or other unexpected 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 a development of assignment 2 with the addition of two additional classes `AudioFrame` and `VideoFrame` that are derived from the abstract base class `Frame` and implement polymorphic inheritance. It is a console application that has an `AnimationManager` that holds a `vector` template of `Animation` objects each of which holds a `forward_list` template of `Frame*` each of which points to either an `AudioFrame` or a `VideoFrame`. Polymorphism is implemented via the pure virtual function `void Frame::CalculateFrameResources()` of the abstract base class. The `string` class is used for all the strings in the application.
Part of the code is shown on the next page; it is also on Brightspace in a text file that you can copy and paste. Because I will use this code when I test your submission, you MUST use this code without modification (not a single character changed): no code added or removed, no new global variables or functions, no new classes, no macros, no defines, no #includes and no statics. Your task is to implement, using C++, only the `AnimationManager`, `Animation`, `Frame`, `AudioFrame` and `VideoFrame` class member functions and the global insertion and extraction operators and not add any new ones.
Everything you write and submit is in the files: AnimationManager.cpp, Animation.cpp, Frame.cpp, AudioFrame.cpp and VideoFrame.cpp. When the application runs you can:

- Add a new `Animation` to the `AnimationManager` at the back of the `vector`
- Delete a particular `Animation`
- Edit a particular `Animation` to maintain its `forward_list` of `Frame*`.
- List the `AnimationManager` to show all its `Animation`s and their `Frames`
- 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:

- dynamic memory management is done with `new` and `delete`
- there is no unused dynamic memory at any time
- input/output is done with `cin` and `cout`
- string objects are used to hold names
- Release of objects' dynamically allocated memory is done in destructors so there is no resource leak (or you lose 30%) – destructors are never explicitly called.

Note that the `forward_list` of `Frame*` are actually pointers to either `AudioFrame` or `VideoFrame` objects. These objects calculate the memory to hold their files using the arrays of constants defined in their classes:

1. AudioFrame: `file size` equals `size / (COMPRESSION_RATIO[i])`
2. `VideoFrame: file size equals size / (COMPRESSION_RATIO*BITDEPTH_FACTOR[i])`

Polymorphism then works through the polymorphic function `CalculateFrameResource()` that is overridden differently for each of them. The essential code that you must use in the overloaded insertion operator friend function of the Animation class to output the compression details of all its frames is:

```
for (it = RA.frames.begin(); it != RA.frames.end(); it++)
      (*it)->CalculateFrameResource()
```

where the actual pointer (`AudioFrame*` or `VideoFrame*`) pointed to is not evident in the code, but the right version of `CalculateFrameResource()` gets called by polymorphism.

See the Marking Sheet for how you can lose marks, but you will lose marks if:
1. [60% penalty] 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, global functions or variables,
2. [>= 60% penalty] It fails to build in Visual Studio 2019,
3. [30% penalty] It crashes in normal operation,
4. It doesn't produce the example output. You should check the input to lie within a valid range and check the correct functionality of all menu items even if they aren't actually shown tested in the example output below.

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 class member functions and friend global functions.

What to Submit : Submit this assignment in the link on the Brightspace site under lab section you are in (Activities/Assignments), as a plain zip file (**not** RAR or 7-Zip or 9 Zip) containing only AnimationManager.cpp, Animation.cpp, Frame.cpp AudioFrame.cpp and VideoFrame.cpp. No other files. 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 tyleraAssign3CST8219.zip. It is also vital that you include file headers (as specified in the Submission Standard) in your source files so they can be identified as yours.

Before you submit the code,

- check that it builds and executes in Visual Studio 2019 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.**

Because of Finals this cannot be late. Don't send me files as an email attachments – they will get 0.

Supplied code (also in a text file on Brightspace you can copy and paste). Don't change it.

```
//Frame.h
#pragma once

class Frame {
         string frameName;
protected:
         double size;
public:
         Frame(string name, double sz) :frameName(name), size(sz) {};
         virtual ~Frame() {};
         virtual void CalculateFrameResource() = 0;
         friend ostream& operator<<(ostream&, Frame&);
};
```

```
//AudioFrame.h
#pragma once

class AudioFrame :public Frame {
          static const int RATES = 3;
          static constexpr double BITRATE[]{128,160,192};
          static constexpr double COMPRESSION_RATIO[]{11.1,9.1,7.1};
          void CalculateFrameResource();
public:
          AudioFrame(string frameName, double fileSize) :Frame(frameName,fileSize) {}
          AudioFrame(const AudioFrame& RA) : Frame(RA) {}
          ~AudioFrame(){}
          friend ostream& operator<<(ostream&, AudioFrame&);
};
```

```
// VideoFrame.h
#pragma once

class VideoFrame :public Frame {
          static const int BITS = 8;
          static constexpr double COMPRESSION_RATIO = 6.0;
          static constexpr double BITDEPTH_FACTOR[]{ 11.1,4.6,3.5,2.4,1.9,1.5,1.2,1.0 };
          void CalculateFrameResource();
public:
          VideoFrame(string fileName, double fileSize ) :Frame(fileName, fileSize) {}
          VideoFrame(const VideoFrame& RV) :Frame(RV) {}
          ~VideoFrame() {}
          friend ostream& operator<<(ostream&, VideoFrame&);
};
```

```
//Animation.h
#pragma once

class Animation
{
```

```
          string animationName;
          forward_list<Frame*> frames;
public:
          Animation(string name):animationName(name) {}
          ~Animation();
          void EditFrame();
          void DeleteFrame();
          friend istream& operator>>(istream&,Animation&);// Add a Frame as in cin >> A;
          friend ostream& operator<<(ostream&,Animation&);// output the Frames as in cout << A;
};
```

```
// AnimationManager.h
#pragma once

class AnimationManager
{
          string managerName;
          vector<Animation> animations;
public:
          AnimationManager(string name) :managerName(name) {}
          ~AnimationManager() {}
          void EditAnimation();
          void DeleteAnimation();
          friend istream& operator>>(istream&, AnimationManager&);// add an Animation
          friend ostream& operator<<(ostream&,AnimationManager&);// output the Animations
};
```

```
// Assignment3.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
#include <crtdbg.h>
#include <iostream>
#include <string>
#include <vector>
#include <forward_list>
using namespace std;
#include "Frame.h"
#include "AudioFrame.h"
#include "VideoFrame.h"
#include "Animation.h"
#include "AnimationManager.h"

int main(void)
{
          char response;
          bool RUNNING = true;
          AnimationManager M("Manager1");
          _CrtSetDbgFlag(_CRTDBG_ALLOC_MEM_DF | _CRTDBG_LEAK_CHECK_DF);

          while (RUNNING)
          {
                    cout<<"MENU\n 1. Add an Animation\n 2. Delete Animation at end\n 3. Edit an Animation\n 4. list the
Animations\n 5. Quit"<<endl;
                    cin>>response;
                    switch (response)
                    {
                    case '1':cin >> M; break;
                    case '2':M.DeleteAnimation(); break;
                    case '3':M.EditAnimation(); break;
                    case '4':cout << M; break;
                    case '5':RUNNING = false; break;
                    default:cout<<"Please enter a valid option"<<endl;
                    }
          }
          return 0;
}
```

Example Output

```
MENU
 1. Add an Animation
 2. Delete an Animation
 3. Edit an Animation
 4. list the Animations
 5. Quit
1
Add an Animation to the Animation Manager
Please enter the Animation Name: Animation1
Animation Animation1 added at the back of animations
MENU
 1. Add an Animation
 2. Delete an Animation
 3. Edit an Animation
 4. list the Animations
 5. Quit
1
Add an Animation to the Animation Manager
Please enter the Animation Name: Animation2
Animation Animation2 added at the back of animations
MENU
 1. Add an Animation
 2. Delete an Animation
```

```
 3. Edit an Animation
 4. list the Animations
 5. Quit
1
Add an Animation to the Animation Manager
Please enter the Animation Name: Animation3
Animation Animation3 added at the back of animations
MENU
 1. Add an Animation
 2. Delete an Animation
 3. Edit an Animation
 4. list the Animations
 5. Quit
3
Which Animation do you wish to edit? Please give the index (from 0 to 2): 0
Editing Animation #0
MENU
 1. Insert a Frame at front
 2. Delete first Frame
 3. Edit a Frame
 4. Quit
1
Insert a Frame in the Animation
Please enter the Frame frameName: Frame1
Please enter the Frame size(MB): 64
 Please enter the Frame type (AudioFrame = A, VideoFrame = V): A
Frame Frame1 Frame* added at the front of frames
MENU
 1. Insert a Frame at front
 2. Delete first Frame
 3. Edit a Frame
 4. Quit
1
Insert a Frame in the Animation
Please enter the Frame frameName: Frame2
Please enter the Frame size(MB): 128
 Please enter the Frame type (AudioFrame = A, VideoFrame = V): V
Frame Frame2 Frame* added at the front of frames
MENU
 1. Insert a Frame at front
 2. Delete first Frame
 3. Edit a Frame
 4. Quit
4
Animation #0 edit complete
MENU
 1. Add an Animation
 2. Delete an Animation
 3. Edit an Animation
 4. list the Animations
 5. Quit
3
Which Animation do you wish to edit? Please give the index (from 0 to 2): 2
Editing Animation #2
MENU
 1. Insert a Frame at front
 2. Delete first Frame
 3. Edit a Frame
 4. Quit
1
Insert a Frame in the Animation
Please enter the Frame frameName: Frame1
Please enter the Frame size(MB): 256
 Please enter the Frame type (AudioFrame = A, VideoFrame = V): V
Frame Frame1 Frame* added at the front of frames
MENU
 1. Insert a Frame at front
 2. Delete first Frame
 3. Edit a Frame
 4. Quit
4
Animation #2 edit complete
MENU
 1. Add an Animation
 2. Delete an Animation
 3. Edit an Animation
 4. list the Animations
 5. Quit
4
AnimationManager: Manager1
Animation: 0
        Animation name is Animation1
        Report the Animation
Frame #0
VideoFrame: frameName = Frame2
Lempel-Ziv-Welch Lossless Compression
-------------------------------------------------------------------------------------------
colours:        | 2      | 4      | 8      | 16     | 32     | 64     | 128    | 256
-------------------------------------------------------------------------------------------
file size (MB): | 1.922  | 4.638  | 6.095  | 8.889  | 11.23  | 14.22  | 17.78  | 21.33
-------------------------------------------------------------------------------------------
Frame #1
AudioFrame: frameName = Frame1
MP3 Lossy Compression
-----------------------------------------------------------
```

```
bitrate (kbits/s):        | 128    | 160    | 192
-----------------------------------------------------------
file size (MB):           | 5.77   | 7.03   | 9.01
-----------------------------------------------------------
Animation: 1
        Animation name is Animation2
        Report the Animation
        No frames in the Animation
Animation: 2
        Animation name is Animation3
        Report the Animation
Frame #0
VideoFrame: frameName = Frame1
Lempel-Ziv-Welch Lossless Compression
-------------------------------------------------------------------------------------------
colours:        | 2      | 4      | 8      | 16     | 32     | 64     | 128    | 256
-------------------------------------------------------------------------------------------
file size (MB): | 3.844  | 9.275  | 12.19  | 17.78  | 22.46  | 28.44  | 35.56  | 42.67
-------------------------------------------------------------------------------------------
MENU
 1. Add an Animation
 2. Delete an Animation
 3. Edit an Animation
 4. list the Animations
 5. Quit
```