Assignment 2 Part 1

Deadline:	**Anytime before Sunday, 3 May 2015, 12 :00 noon**
Evaluation:	10 marks – which is 5% of your final grade
Late Submission:	1 mark off for every **8 hours late (**so 80 hours late gets 10 marks deducted).
Purpose:	Practice with inheritance and polymorphism.

Problem to solve: You have to write a program to track movies, including specialized movies such as foreign movies and revised versions (versions updated by the director after the initial release). You have to use inheritance. Since every movie has a title, director, running time, and quality (0 to 4 stars) class **Movie** is the base class with data members to hold the common information, methods to read and store information in the data members, and a method to output the data. Class **Revised** is derived from **Movie**. This class includes data members to hold the revised time and the changes, methods to store the added information in the data members and a method to output the data. Finally, class **Foreign** is derived from class **Movie** too. It contains a data member to hold the language, a method to store the language in the data member and a method to output the data. Therefore you have to design and implement three classes: Movie, Revised and Foreign. **Write all your code in a file named *A2P1.h***

Requirements:
Here is a recommended minimal interface you have to provide for the Movie class

```
class Movie {
public:
   Movie();
   virtual ~Movie(){}
   void setTitle( const string& t);
   void setDirector( const string& d );
   void setTime( int t = 0 );
   void setQuality( int q = 0 );
   virtual void display() const;
   virtual void getData( ifstream& fin);
   static bool readFromFile( string fname, Movie* Movies[ ],int n);
private:
   string title;
   string director;
   int time;   // in minutes
   int quality; // 0 (bad) to 4 (tops)
};
```

Provide a polymorphic input method **`getData()`** that reads records for Movies, Revised, and Foreign from an input file (*inF.txt*). Each set of records in the input file represents an object in the hierarchy. The input file contains information (records) about movies. Each record falls into one group. The 1st record in each group is a line-string that represents a Movie type: "Movie", "Foreign", "Revised", the title is on the next line, the director's name(s) is on the next line, the minutes representing the running time is an integer contained on the next line and next line contains an integer (0 to 4) for the quality.
For the revised movies the next two items are the revised running time (an integer) and a string describing the changes that were done while for the foreign movies the extra information contained in the input file is the language. Assume that the input file is correctly formatted (see Fig 1 for an example).

inF.txt - Notepad

File Edit Format View Help

```
Movie
Last Tango in Paris
Roman Polanski
97
3
Movie
Casablanca
Dick Bogard
121
4
Foreign
SpaceToon
Mickey Mouse
90
4
Italian
Revised
Peace and War
Nikolai Pavlov
134
3
123
War scene has been shortened.
Movie
Home alone
Pink Panther
87
5
Movie
Another Last Tango in Paris
Bla Polanski
97
3
Movie
Guarding Tess
I Do Not Know
121
4
Foreign
Vive la France
Jean Marais
90
4
French
Revised
Brave Hearth
Mel Gibson
134
3
211
More war scenes have been added.
Movie
The Last Athenian
Bingo Bonzo Bla Bla
87
5
```

Fig 1-Input file ***inF.txt***

main

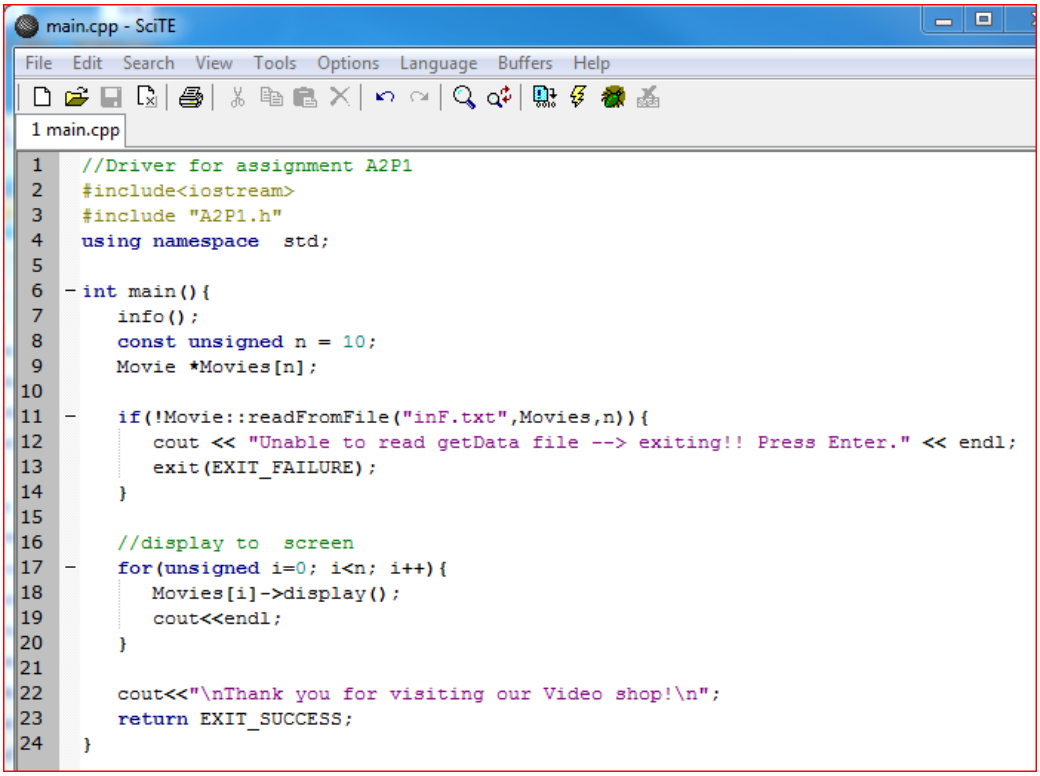
```
159.234 Assignment 2 P1
CALUDE E ID 0987654
BUNNY E ID 9987654
-------------------

Title: Last Tango in Paris
Director: Roman Polanski
Time: 97 mins
Quality: ***

Title: Casablanca
Director: Dick Bogard
Time: 121 mins
Quality: ****

Title: SpaceToon
Director: Mickey Mouse
Time: 90 mins
Quality: ****
Language: Italian

Title: Peace and War
Director: Nikolai Pavlov
Time: 134 mins
Quality: ***
Revised time: 123
Changes: War scene has been shortened.

Title: Home alone
Director: Pink Panther
Time: 87 mins
Quality: *****

Title: Another Last Tango in Paris
Director: Bla Polanski
Time: 97 mins
Quality: ***

Title: Guarding Tess
Director: I Do Not Know
Time: 121 mins
Quality: ****

Title: Vive la France
Director: Jean Marais
Time: 90 mins
Quality: ****
Language: French

Title: Brave Hearth
Director: Mel Gibson
Time: 134 mins
Quality: ***
Revised time: 211
Changes: More war scenes have been added.

Title: The Last Athenian
Director: Bingo Bonzo Bla Bla
Time: 87 mins
Quality: *****


Thank you for visiting our Video shop!
```

Fig 2. Output produced by the program in Fig 3.

Make the `getData()` method in the Movie hierarchy polymorphic. This is used (by `readFromFile()` methode) to read, from the file *inF.txt*, data for each dynamically created object in the hierarchy.

Provide a polymorphic method, **`display(…)`**, which will display on screen, data for each dynamically created object.

Most methods are self-explanatory but the method `readFromFile()` needs some clarifications. The method reads data from an input file, dynamically creating the appropriate Movie object for each record group. For instance, a Foreign object is dynamically created if data represent a Foreign movie rather than a regular movie or a revised movie. Pointers to dynamically created objects are stored in the array Movies of size n. Returns true to signal success and false to signal failure.
See Fig 2 for the output produced when the driver program in Fig 3 was used.




```
//Driver for assignment A2P1
#include<iostream>
#include "A2P1.h"
using namespace  std;

int main(){
   info();
   const unsigned n = 10;
   Movie *Movies[n];

   if(!Movie::readFromFile("inF.txt",Movies,n)){
      cout << "Unable to read getData file --> exiting!! Press Enter." << endl;
      exit(EXIT_FAILURE);
   }

   //display to  screen
   for(unsigned i=0; i<n; i++){
      Movies[i]->display();
      cout<<endl;
   }

   cout<<"\nThank you for visiting our Video shop!\n";
   return EXIT_SUCCESS;
}
```

Fig 3 Driver program

Hand-in: Submit for marking your **A2P1.h file** via Stream.

Miscellaneous:

1. Programs that do not compile in the lab, using gcc, get 0 marks.
2. Marks will be allocated for: correctness, completeness, clear and simple design, use of C++ constructs, documentation, and clear output.
3. Using goto, non-constant global variables, C-like I/O constructs (i.e printf, fprintf, scanf, FILE*, etc) is not allowed and it will be penalized by marks deduction.
4. When working in teams (at most two students per team), send one solution file per team.
5. The assignment will be previewed on Wednesday lecture before it's due.

If you have any questions about this assignment, please ask the lecturer before its due time!