CSC 4111 Winter 2018 Lab 1

This lab assignment is due 17/01/2018 at **7:00pm**. See the syllabus for late penalties.

You should turn it in by blackboard. Each task should be in a separate folder named **taskN**, where N is the task number, inside a zipped file named: **YourLastName_YourFirstName.zip**.

You may ask other members of the class questions about the assignment. However, you must do your own assignment; you may not use anyone else's code. **You must write your own code.** Also, any coding task that doesn't compile will receive 0.

Task 1 - 40 points
Write a C++ class (or a set of classes) that implements a stack using a linked list. The type of data contained in the stack should be int. The maximum size of the stack is 20. Implement at least the following methods:

- Constructor and destructor; // 5 pts
- **void push (int number)**; // pushes an element with the value number into the stack. 5 pts.
- **int pop ()**; // pops an element from the stack and returns its value. 5 pts.
- **int isEmpty();** // returns 1 if the stack is empty, 0 otherwise. 5 pts.
- **int numOfElements();** // returns the number of elements in the stack. 5 pts.
- **void printElements();** // print out the current stack to the console. 5 pts.
- A main function; // execute each method of your stack (except the destructor) at least once. 10 pts.

Remove the debug folder from the self-contained project and then put the entire project in a folder named Task1.

Task 2 – 30 points
We need to write a program to help our favorite team keep track of its players' performance. The file that will print out the player's stats is done. You are given a class for a player; however, a football team is composed of offensive and defensive players. So you need to build on the player class. It is rare that a player is both, so we will assume our team has exclusively offensive and defensive players. All players have a name, number and they all play for some number of minutes. However, a defensive player is measured by the number of tackles they get, while an offensive player is measured by the number of yards they get. For this assignment you need to create 2 classes that inherit from a class player that has already been written. This class makes no calculations; it just holds data on each player. Your class will be used by the main function in main.cpp to print the team's stats, so it must follow the guidelines below. You may not change the files given in any way.

Download the Football project and unzip it. It does not compile because it misses some classes. **The project needs 4 files added: defense.h, defense.cpp, offense.h, offense.cpp (10 points). There are also a few questions that must be answered (20 points).**

Note the main.cpp, player.h and player.cpp files must not be changed in any way. 10 points will be deducted if any of them has been modified.

The added files should be formatted similarly to the Player class files. The classes need to have the following, and you must decide the return types, visibility and any additional modidifiers for these methods:

defense class fields:

```
int tackles;
```

defense class methods:

```
Defense(string name);
setMinutesPlayed(int minutes);
setTackles(int tackles);
printStats() const;
```

offense class fields:

```
int yards;
```

offense class methods:

```
Offense(string name);
setMinutesPlayed(int minutes);
setYards(int yards);
printStats() const;
```

Rules:

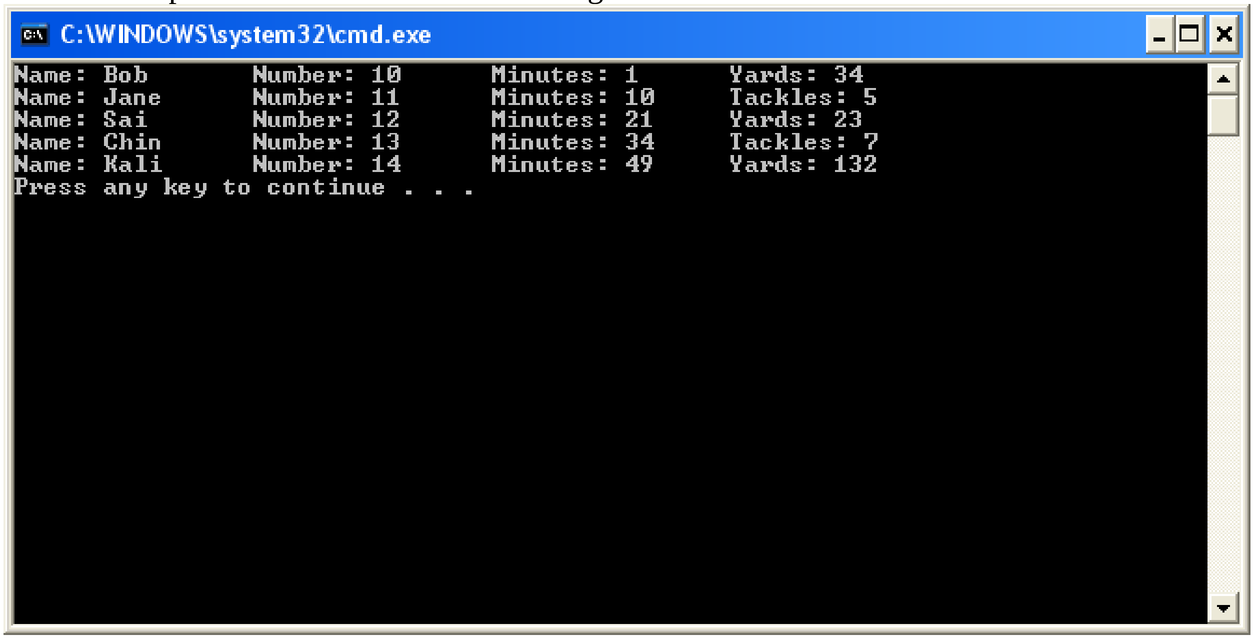
- Your classes must inherit from the base class Player.
- You should not recode anything. If the base class already does something, do NOT put it in your derived class. Use what has been already there.
- You should follow the coding conventions of the program given.
- All files should have a brief description at the beginning that includes the title of the file and your name. The code should have comments describing each method and large section. See the provided code for examples.
- All 4 of the methods listed for each new class above must be implemented. You can add extra methods, but it is not necessary and there is no bonus.
- This is just a suggestion: the solution averages exactly one line of code (LOC) inside each method, so if a method has 2 LOC, another should have 0 LOC. There is no trick here either with crazy lines that call multiple methods or anything else. Basically, if you are writing many lines of code per method, you need to review inheritance or polymorphism.
- All classes should include all needed libraries.

Questions:

1) On lines 29 to 33 of main.cpp, why is there an '&' before the player variables?
2) For lines 45 to 49 of main.cpp, why not use a loop?
3) On lines 10, 11 and 52 of player.h, what do the #ifndef, #define and #endif preprocessor commands do?

4) On line 20 of player.h, the string name is private, what did you have to do because of this?
5) On lines 24 of player.h, why does minutes played have to be protected, not private?
6) On line 43 of player.h, why is the function assigned zero?
7) On line 49 of player.h, what does the word 'const' do?
8) What method should be entirely in the base class? Why?
9) Which class is an abstract class? Why?
10) Name one thing you learned or had forgotten and remembered in this task.

The final output should look like the following:

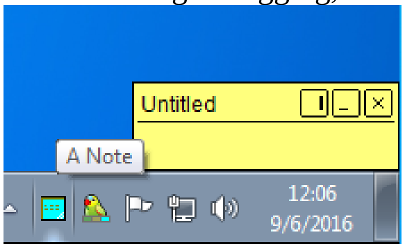


```
Name: Bob       Number: 10      Minutes: 1      Yards: 34
Name: Jane      Number: 11      Minutes: 10     Tackles: 5
Name: Sai       Number: 12      Minutes: 21     Yards: 23
Name: Chin      Number: 13      Minutes: 34     Tackles: 7
Name: Kali      Number: 14      Minutes: 49     Yards: 132
Press any key to continue . . .
```

Remove the debug folder from the project. Answer the 10 questions in a text file, then put it with the cleaned project in a folder named Task2.

Task 3 – 10 points

Download and open ANote project, then build it. There are compilation errors. Fix them. Once you solve the bug of ANote, you should see the project running in the notification area of the taskbar during debugging, as shown below:



Answer how to fix those errors (which files need to change, the deleted/modified/added code, etc.) in a text file, then put it in a folder named Task3**. Do NOT submit the fixed project/program.**

Place all 3 folders into a zipped folder called **YourLastName_YourFirstName.zip** and turn in**.**

Only the top folder should be zipped. Upload it to blackboard, then download it yourself to ensure the submission is correct. If your file is not uploaded correctly, you will be subject to the late policy.

In CSC 4111, for each assignment/project, I have/will have specified what to submit (source code, a single file, etc.). Submit redundant files (e.g. upload program when only a single file is required) will lose 10 points for each corresponding task.