**COS-213: C++ PROGRAMMING**

**Your final project includes five problems.**

**Submit your code files to your mentor. Submit the code for each problem separately. The problems vary in complexity and so have different weightings for your total project grade.**

**Important: Please place all appropriate coding comments inside of the code.**

## Final Project Problem 1

*This problem counts for approximately 30 percent of the final project grade.*

This assignment requires you to code a simulation race between the tortoise and the hare. For this project, you will be using random-number generation to move the creatures. To make things more interesting, the animals have to race up the side of a slippery mountain, which could cause them to lose ground. In this race either animal could win or there could be a tie with no clear winner.

The animals begin at "Square 1" of 70 squares. Each of the squares represents a position the animal can hold along the racetrack. The finish line is at Square 70. When an animal wins, a winning message of your choice should be posted. For example:

- Yay! The rabbit won! He hops the fastest!
- Woo-hooo! Slow and steady wins the race! Congratulations, turtle!
- Tie score–no winner! Want to race again?

To start the race, print a message similar to:

- Bang! Off they go!

There is a clock that ticks once per second. With each tick of the clock, your program should adjust the position of the animals according to the following rules:

| Animal | Move Type | Percentage of the Time | Actual Move |
|---|---|---|---|
| Tortoise | Fast Plod | 50% | 3 squares to the right |
| Tortoise | Slip | 20% | 6 squares to the left |
| Tortoise | Slow Plod | 30% | 1 squares to the right |
| Hare | Sleep | 20% | No move at all |
| Hare | Big Hop | 20% | 9 squares to the right |
| Hare | Big Slip | 10% | 12 squares to the left |

| | | | |
|---|---|---|---|
| Hare | Small Hop | 30% | 1 square to the right |
| Hare | Small Slip | 20% | 2 squares to the left |

Keep track of the positions of the animals by using variables. If an animal slips, the lowest it can go is back to position 1. The highest it can go is to position 70 when it wins the race.

You will work with the percentages in the table above by generating a random integer current in the range 1 <current ≤ 10.

For the tortoise, a "fast plod" is when 1 ≤ current ≤ 5, a "slip" when 6 ≤ current ≤ 7, or a "slow plod" 8 ≤ current ≤ 10. A similar approach would be used to set up the moves for the hare.

For each tick of the clock (each repetition of the loop), print a 70-position line showing the letter T in the tortoise's position and the letter H in the hare's position. If the two animals land on the same square, which may happen, the animals will bump. If this happens, print BUMP! at the current position. (There is no Bump penalty for either animal.)

After you print each line, check to see if either animal has landed on Square 70. If this happens, print a winning-type message.

It will make the simulation more interesting if you have the user press any key after each iteration of the loop, so that they can see the movement of the animals.

Output: You should develop the output for this assignment to be as clear and concise as possible. Provide as much detail as possible in regards to the character and related character move.

**Final Project Problem 2**

*This problem counts for approximately 30 percent of the final project grade.*

Create a *class* that simulates an alarm clock. In this *class* you should:

- Store time in hours, minutes, and seconds. Note if time is AM or PM. (Hint: You should have separate private members for the alarm and the clock. Do not forget to have a character variable representing AM or PM.)
- Initialize the clock to a specified time.
- Allow the clock to increment to the next second. (Hint: You need to take into account things like if the clock's time is 11:59:59 AM and you increment by a second, the time will be 12:00:00 PM. You may need to consider some iterated *if* statements.)
- Set the alarm and have the alarm print out "WAKE UP" when the set time is reached. (Hint: You may wish to create a private function that provides the wished-for printout when the alarm time is reached and the alarm clock is on.)

- Display the present time.
- Use the class in a program that uses the functions requiring displaying of time and setting of the alarm.

Include 2 constructors. One constructor should be the default constructor that will initialize the object to 12:00:00 AM. The second constructor should take parameters for hours, minutes, seconds, and AM/PM. Both constructors will provide the private members with the time. In addition, have both constructors set the alarm clock as off. (You will need a Boolean variable that determines whether the alarm is on or off). The function or method you use to set the alarm will set the alarm on.

Output: Provide a detailed description and display of the values that have been created as variables are moved through the class. Display a data trace for this assignment.

**Final Project Problem 3**

*This problem counts for approximately 15 percent of the final project grade.*

Write a program that reads from the external file input.txt, counts the letters in every word, replaces the word by that number, and then writes the numbers to an external file output.txt (Note: Do not forget to copy the blanks. You may wish to use infile.get and outfile.put in your program.) Also you may wish to use the strlen( ) function.

Output: After the program generates output.txt, the code should display the contents of the file on the screen to verification.

**Final Project Problem 4**

*This problem counts for approximately 10 percent of the final project grade.*

Create a template that changes the value of 2 variables. (Hint: Use reference parameters since you will swap the variables within the function.) Use the template in a program where you change the value of 2 variables of type *int* and 2 variables of type *long.*

Output: After the program generates ooutput.txt, the code should display the contents of the file on the screen to verification.

**Final Project Problem 5**

*This problem counts for approximately 15 percent of the final project grade.*

Derive the *cube* class from the base *square* class. Assume the *square* class has a protected member variable representing the side called *side* and declared as a *double* with a default value of 1.0. It also has a public function called *calcVal* that evaluates the area of a square as side * side.

In your derived class have the default values for *side* be 1. For the *cube* class include a public function *calcVal* that evaluates the volume of the *cube*. (Hint: The volume of the cube is side * square :: calcVal.)

Output: Provide a detailed description of the process that took place in developing the solution to the above-mentioned question.