Object Oriented Programming

In this assignment, you are going to implement some classes which are integrated in a large system. You can see the files which can be found in 00\_StudentWork. These classes are:

mySystem\_MonteCarlo

mySystem\_SineCosineFunction

mySystem\_CubicFunction

mySystem\_StudentManager

**Write your programs in .NET2010. You must use the assignment template to implement your programs.**

**We will rebuild your program in the Release mode and check your program.**

**You can find the executable file**

**enjoy\_programming.exe**

**in ./bin/Release**

**The demo program may have bugs. Thus, please follow the instructions. If the instructions are not clear to you, send us an email to clarify the problems.**

Requirement Specification

Use double to define a variable which is a floating point number. All the calculations should be done in double precision. Don’t use float. Show the value of a floating point number up to 8 decimal digits.

1. **Basic tasks.**
2. **Write your name in the header file mySystemApp.h**
3. Press ‘s’ or ‘S’ to show your student information: student ID, name and email address. showMyStudentInfo( ) in mySystemApp.cpp
4. Set STUDENT\_INFO for your name and student ID in mySystemApp.cpp.

**Items I, II and III must be done. If not, your score is zero.**

Key usages:

F1: perform Monte Carlo Simulation

F2: perform Sine-Cosine Function Calculation

F3: perform Cubic Function Calculation

F4: perform the student record management

i, I: ask for input

s, S: show the student information

**In mySystem\_MonteCarlo, implement the followings.**

1. Show a message about the Monte Carlo simulation.
2. Press ‘i’ or ‘I’ to ask the user to input the radius of a circle. The radius is inside [1,10].
3. Ask the user to input the number of samples. The number of samples is between 1 and 1,000,000.
4. Use the Monte Carlo simulation to compute the pi value and show the pi value. Use generateUniformSample ( ) to generate samples. Use computePI( ) to compute the value of pi and to output pi.
5. Get the radius getRadius( ).
6. Get the number of samples getNumSamples( ). Implement getSample( … ).
7. Implement reset( ) to recompute the samples and pi.
8. Get the coordinates of a sample based on the sample index (starting from 0). Return true if the sample lays inside the circle. Otherwise return false.
9. Press ‘<’ to decrease the number of sampler points by 2000 each time. Compute the new samples. Implement method decrease( )**.**
10. Press ‘>’ to increase the number of sampler points by 2000 each time. Compute the new samples. Implement method increase( ).
11. Press ‘n’ to decrease the radius of the circle by 1. The smallest radius is 1. Compute the new samples and output pi**.** Implement method decreaseRadius( )**.**
12. Press ‘m’ to increase the the radius of the circle by 1. The smallest radius is 1. Compute the new samples and output pi**.** Implement method increaseRadius( )**.**

**In mySystem\_SineCosineFunction, implement the following items.**

1. The default number of sample points is 100.
2. Press ‘i’ to ask the user to input the three coefficients: a and b.
3. Press ‘i’ to ask the user to input the range of x, i.e., minimum value and maximum value of x. The minimum and maximum values should be in the interval [-10, 10].
4. Press ‘i’ to ask the user to input the number of sample points. Interval [2, 100]
5. Get the range of x.
6. Get the number of sample points.
7. Get the value of the function for a given x value.
8. Press ‘<’ to decrease value a by 0.05.
9. Press ‘>’ to increase value a by 0. 05.
10. Press ‘1’ to decrease value b by 0. 05.
11. Press ‘2’ to increase value b by 0. 05.
12. Press ‘3’ to decrease the number of samples by 5. The minimum number of samples is 10.
13. Press ‘4’ to increase the number of samples by 5.

The function is: f(x) = a cos (b x)

Do some experiments to have fun!

**In mySystem\_CubicFunction, implement the following items.**

1. Ask the user to input the four coefficients: a, b, c and d.
2. Ask the user to input the range of x, i.e., minimum value and maximum value of x. The minimum and maximum values should be in the interval [-100, 100].
3. Ask the user to input the number of sample points. Interval [2, 100]
4. Get the range of x.
5. Get the number of sample points.
6. Get the value of the function for a given x value.
7. Press ‘<’ to decrease value a by 0.001.
8. Press ‘>’ to increase value a by 0. 001.

The cubic function is: f(x) = ax3 + bx2 + cx + d

Play with it. Press ‘<’ and ‘>’.

**In mySystem\_StudentManager, implement the following items.**

1. Press ‘i’ or ‘I’ to ask for input.
2. Ask the user to input the number of students. The number of students is in [2,100].
3. Ask the user to input the score of each student. The score range is [0, 100].
4. Show the range of the scores.
5. Show the average score.
6. Show the standard deviation of the scores.

Standard deviation = sqrt( sum(x – x’)\*( x – x’)/(n-1) ), for scores of all students, where x is the score of a student and x’ is the average. Read the article about standard deviation in Wiki if you are not sure what it is.

1. Show the scores in an ascending order. That is, sort the scores.
2. Get the number of students whose score is inside an interval [s0, s1] (inclusive). Implement getNumOfStudentsInScoreInterval(…).

The program shows the histogram of the scores of all the students.

Use ( rand( ))/(double) (RAND\_MAX)

to compute a random value between [0, 1].

**Submission**:

1. Change the folder name to ID\_Name, where ID is your student ID and Name is your name. Upload the entire folder of the source code to E3 platform before the deadline.
2. You must demo your work to our TAs in the lab session.
3. **If you cannot demo your programs, your score is zero.**

**Penalties:**

1. **Late submission: 40% penalty each day.**
2. **Cheating: you will be received a score of zero, e.g., borrowing your source code to others or/and copying others’ source code.**

**The folder 00\_StudentWork stores the files.**

