MTH 3300
Spring 2017
E. Fink
March 5, 2017

Homework 5

Directions: Write the following 2 programs, and submit your source code to me via Blackboard, using the template files I have provided. You should send only the source code, in cpp format; do NOT send your .exe files. READ THE INSTRUCTIONS on how to submit your work in the Course Documents section of Blackboard.

This programming assignment is meant to give practice with `if-else`, loops, file-handling, and random numbers.

1) payroll.cpp

Consider the file `timesheet.txt`. Each row contains, separated by spaces:

- an employee's first name;
- the employee's last name;
- the number of hours worked in a week for each of two weeks; and
- the employee's normal pay rate, expressed in dollars per hour.

So, for example, John Adams worked for 30 hours the first week, and 55 hours the second, and gets paid $10 per hour.

Create a program that reads in this file and then outputs the employee's total pay for the two weeks to a file called `checks.txt`. Your program should work for ANY file that is called `timesheet.txt`, as long as it contains 5 rows, in the format described above.

Additionally, KEEP IN MIND the following rule: if an employee works for more than 40 hours in a week, they get overtime pay for the overtime hours, which is 1.5 times regular pay. So, in my example, Adams gets a paycheck of

$$\$10 \times 30 + \$10 \times 40 + \$10 \times 1.5 \times 15 = \$925$$

(30 hours of regular pay in the first week, 40 hours of regular pay in the second week, 15 hours of time-and-a-half in the second week).

Furthermore, the output file `checks.txt` should look like the example file provided: the newlines and number of digits in the paycheck amounts should be as shown.

I would use a loop, although if you like, I will not deduct any points for avoiding them.

Specifications: your program must

- be able to open and read any file named `timesheet.txt` that is placed in the correct directory and has the same FORMAT as the sample one I provided (5 rows, each with data ordered as above) – your program should NOT reflect any of the specific names or data that I provided in my example.
- create a file named `checks.txt` that contains 5 rows, each containing a name followed by a check value (again, formatted like my sample output).
- calculate pay correctly, including the overtime rule.

2) exam.cpp

You forgot to study for the final exam! Oh no!

There's some good news and some bad news. The good news is that the exam is multiple choice: 35 questions, each with 5 choices. So you have some chance at guessing correctly. The bad news is that you have to get at least 11 questions correct to pass the class. (Obviously, this final exam is too easy to be given by the math department.)

What is the probability that you pass?

Write a program to estimate this probability, by simulating 1,000,000 exams. Specifically, the following should be executed in a loop, which executes a million times:

> To represent a student's attempt at taking the exam, the program should generate 35 random numbers, each between 1 and 5. We will let the value 5 represent a correct answer – after all, the probability of getting a 5 is the same as the probability of guessing correctly (20%). The program should tally the number of correct answers, and decide if this exam is a pass or a fail, based on whether there are ≥ 11 correct answers or not.

The probability should be approximately equal to $\frac{\text{number of passes}}{\text{number of trials}} = \frac{\text{number of passes}}{1000000}$. This value is what your program should print out.

NOTE: if your program is working correctly, you should get DIFFERENT answers each time you run it, although the probability should be close to the same each time you run it.

Specifications: your program must

- NOT ask the user for any input.
- print out an estimate of the probability that a student blindly guessing on a 35 question multiple choice exam, each question with 5 answer choices, gets at least 11 questions correct.
- **use a simulation with random numbers – no credit for theoretical solutions**. If you get the point of this assignment, it should be clear that your answers should **not** be exactly the true probabilities, and that it should **not** even give the same answer each time you run it.