APS 105 — Computer Fundamentals

Lab #2: Programs That Calculate

Winter 2021

Important note: You must submit your solution **by 11:59 p.m. on the day of your lab session.**

Objective

In this lab you will be writing three short programs using variables and arithmetic operators. Input and output must be done using scanf and printf. You will make use of the standard I/O and math libraries.

In your programs, you may only use the programming techniques presented in lectures up to this point. It is highly recommended that you document your program with comments. Your comments should at least include your full name and student number, and a very short description of what the program does.

Sample output

In the sample output examples that follow, the text that would be entered by the program user is displayed using a **bold** font. The text <**enter**> stands for the user pressing the enter key on the keyboard. On some keyboards, the enter key may be labeled return. When you execute your programs, the user's text will not be displayed in bold and <enter> will not be shown.

Part 1 — Cash back

A store is having a promotion advertised as "for every 3 dollars spent on produce in the store, you get 10 cents cash back". Write a C program that helps you to calculate the total amount you will pay. Your program reads in the price per pound of the produce, the weight of the produce (in pounds) and outputs the total charge and the total savings **rounded to two decimal places**.

Here is sample output from an execution of the program:

```
Enter the price per pound:  2.99<enter>
Enter the total weight:  10.5<enter>
The total charge is:  30.40
You saved:  1.00
```

Note: You can assume that the user enters valid numbers. You should assume that there are no taxes. In your solution, you may find useful calling the function floor from the math library, which takes a floating point number, x, and returns the largest integer that is not greater than x. Alternative implementations are also possible.

Your C program must go in a file named Lab2Part1.c.

Part 2 — Price Rounding

When you make a purchase at the grocery store, the price you pay needs to be rounded to the closest nickel. Write a C program that helps the cashier round the price to the closest nickel.

```
Enter the price as a float with two decimals: 2.99<enter>
The total charge is: 3.00
```

```
Enter the price as a float with two decimals: 2.92<enter>
The total charge is: 2.90
```

Note: You can assume that the user enters valid numbers. In your solution, you may find useful calling the function round from the math library, which takes a floating point number, x, and returns the integer that is closest to the argument.

Your C program must go in a file named Lab2Part2.c.

Part 3 – Interest Compounding

In order to keep our money safe we usually keep it in a bank. A savings account in a bank not only keeps our money safe but gives us an interest rate, so that our money can grow over time. The interest rate is given as a percentage of our money over a time unit, e.g., 1% per year, which can be computed and added, to the money in the account, called compounding, at various intervals within the time unit, which is a year. The formula by which our money grows, if we consider that the amount we deposit initially is P and the interest rate is r, over t time units, and the number of times that the interest is compounded during the time unit is n is as follows:

$$A = P(1 + \frac{r}{n})^{nt}$$

Design and write a program which computes the projected money in the account A, given various P, n, t, and r.

Your program reads **P**, the initial deposit amount, also called the principal, **r,** the annual interest rate (floating point with two decimal places), **n**, the number of times that interest is compounded per year e.g., daily or monthly, and, **t**, the number of years the money is invested (assume whole years). It calculates **A**, the future value of the investment after t years, including interest, rounded to two decimal places.

If an amount of P is deposited into a savings account at an annual interest rate of r, **compounded n times a year**, the value of the investment after t years can be calculated, rounded to two decimal places, by the formula above.

```
The amount deposited P: 5000<enter>
The interest rate r: 0.05<enter>
The number of times the interest is compounded n: 12<enter>
The period of time t the money is invested (in years): 10<enter>
The future value of the investment A: 8235.05
```

Note: To compute x^y, use the pow function in the math.h library. To use the math.h library, you will need to add the following line to the beginning of your program: #include <math.h>.

Your C program must go in a file named Lab2Part3.c.

1 Grading by TA and Submitting Your Program for Auto-Marking

There are a total of 10 marks available in this lab, marked in two different ways:

1. By your TA, for 4 marks out of 10. Once you are ready, show your program to your TA so that we can mark your program for style, and to ask you a few questions to test your understanding of what is happening. Programs with good style have been described in Lab 1, but once again, they are:

   - Clear comments that describe what is happening in the program.
   - Good choices for variable names that indicate their purpose. Please adopt the naming convention where if you have a variable that is described by multiple words, user lower case for the first letter of the first word, and Upper case for all subsequent words e.g. inputCode.
   - Properly indented code.
   - Proper use of named constants, rather than putting constants (such as 125) directly into the code.

   The TA will also ask you some questions to be sure that you understand the underlying concepts being exercised in this lab.

2. By an auto-marking program for 6 marks out of 10. You must sumbit all of your program filesthrough the ECF computers for marking. We will use a software program to compile and run your program, and test it with different inputs. Long before you submit your program for marking, you should run the exercise program that compiles and runs your program and gives it sample inputs, and checks that the outputs are correct. Similar to Lab 1, you should run the following command:

   ```
   /share/copy/aps105s/lab2/exercise
   ```

   within the directory that contains both your solution programs. This program will look for the files comprising Lab2 in your directory, compile them, and run them on some of the test cases that will be used to mark your program automatically later. If there is anything wrong, the exercise program will report this to you, so read its output carefully, and fix the errors that it reports.

   IMPORTANT: YOU WILL HAVE TO COPY ALL OF YOUR FILES TO BE IN THE SAME FOLDER/DIRECTORY WHERE YOU WILL RUN THE SUBMIT COMMAND.

3. Once you have determined that your program is as correct as you can make it, then you must submit your program for auto-marking. This must be done by the end of your lab period as that is the due time. To do so, go into the directory containing your solution files and type the following command:

   ```
   /share/copy/aps105s/lab2/submit
   ```

   This command will re-run the exercise program to check that everything looks fine. If it finds a problem, it will ask you if you are sure that you want to submit. Note that you may submit your work as many times as you want prior to the deadline; only the most recent submission is marked. All files to be submitted as solutions to the lab must be within the same directory, and the submission script must be run also *from that directory via the command line*.

   The exercise program (and the marker program that you will run after the final deadline) will be looking for the exact letters as described in the output in this handout, including the capitalization. When you test your program using the exercise program, you will see that it is expecting the output to be exactly this, so you will have to use it to see if you have this output correct.

   Important Note: You must submit your lab by 11:59 p.m. after the end of your assigned lab period. Late submissions will not be accepted, and you will receive a grade of zero.

You can also check to see if what you think you have submitted is actually there, for peace of mind, using the following command:

```
/share/copy/aps105s/lab2/viewsubmitted
```

This command will download into the directory you run it in, a copy of all of the files that have been submitted. If you already have files of that same name in your directory, these files will be renamed with a number added to the end of the filename.

2 After the Final Deadline Obtaining Automark

Briefly after all lab sections have finished you will be able to run the automarker to determine the automarked fraction of your grade on the code you have submitted. To do so, run the following command:

```
/share/copy/aps105s/lab2/marker00
```

This command will compile and run your code, and test it with all of the test cases used to determine the automark grade. You will be able to see those test cases output and what went right or wrong.