COMPSCI 2211b
Software Tools and Systems Programming

Assignment #3

C Programming

Posted:	February 28th 2018
Due:	March 14th 2018 11:55PM
Total:	100 Points (5% of Final Grade)

Learning Outcomes

By completing this assignment, you will gain and demonstrate skills relating to:

- Writing, compiling and running C programs.
- Use of input/output functions (`printf` & `scanf`)
- Practice with loops, conditional statements and functions.
- Documenting and commenting C source code.

Instructions

For this assignment, an electronic submission is required through OWL. This submission should include the source code for each of the C programs your are required to create. These files should be named as per the instructions given in each question and end in a .c file extension. The files should be attached individually, and not contained in an archive (e.g. zip, tar, etc.) or other binary format (e.g. they should not be in a word or PDF document). You can copy your files off the course server using the `scp` command on the lab computers or using a UNIX-Like operation system. On Windows you can use a program like WinSCP to copy files to/from the course server.

All C programs must work on the course server. Programs will be marked on the course sever and it is expected that students at least test their code on the server before submitting. It will be assumed that the GNU89 C standard (the default for `gcc` on the course server) is being used unless you specifically state otherwise in a comment on the first line of your C source code. Your code must work with the GCC compiler on the course server and you may only use C standards that are supported by this compiler. C++ code or code written in other programming languages will be given 0 marks.

You must document your code. It is expected that you document your C source code with comments. Each program should have comments at the top of the file that state what C standard you are using (if different from the default), any special instructions for compiling/running your code, your name, student number and a short description of what the program does. You are also expected to include in-line comments throughout your code to make each line understandable by anyone who reads it. Lines that are self explanatory or straightforward do not need to be documented.

You will be assessed on the following:

- Completion of each question correctly.
- Providing your source code as required in the above instructions.
- Documenting your code with comments as required in the above instructions.
- Ensuring that your code works on the course server.
- Naming your files correctly and submitting via OWL.

Questions

Question 1 (23 Marks)

Create a C program named ***phone.c*** that takes as input (over standard input) a 10 digit phone number and a single character. The format of the phone number will be `DDDDDDDDDD`, that is, 10 digits in a row with no spaces or other characters. The character will be a single uppercase or lowercase letter between A and D (e.g. A, b, c, D, d, etc).

Your program will display a menu of options and ask the user for input as shown in the Example Input/Output below (at the end of this question) and format the number based on the menu option (character) input by the user. The following table describes how to format the phone number for each menu option:

Menu Option	**Phone Number Format**
A or **a**	**Local format:** `DDD-DDDD` omit the area code (first 3 numbers). For example, 5199145555 would become 914-5555.
B or **b**	**Domestic:** `(DDD) DDD-DDDD` For example, 5199145555 would become (519) 914-5555.
C or **c**	**International:** `+1-DDD-DDD-DDDD` assume country code is 1. For example, 5199145555 would become +1-519-914-5555.
D or **d**	**Odd:** `ODDD ODDD DDDD`, use the minimum with specifier to output each part of the phone number with at least 6 spaces and use the precision specifier to output each number with at least 4 digits. Do not pad the numbers manually (allow the specifier and printf to do this for you). For example, 5199145555 would become ••0519••0914••5555 (spaces added by the minimum width specifier shown as •, red numbers added by precision specifier).
Any other character	Print an error message and exit with an exit status that indicates failure.

You are required to do error checking to test for invalid input. You must check the return of `scanf` and print an error if it aborted early. You must also check that no part of the input number is negative and that the input option is valid. If an error is encountered print an error message and exit with a non-zero exit status (to denote failure), otherwise exit with an exit status of 0 (to denote success).

Hint: *You do not need to use strings, arrays, pointers or loops for this question. If you are, you are likely over thinking it.*

Example Input/Output (input in green):

Example 1:

```
Input Phone Number:  5551234567

Format Options:
A) Local
B) Domestic
C) International
D) Odd

Input Option:  A

Phone Number:  123-4567
```

Example 2:

```
Input Phone Number:  5551234567

Format Options:
A) Local
B) Domestic
C) International
D) Odd

Input Option:  B

Phone Number:  (555) 123-4567
```

Example 3:

```
Input Phone Number:  5551234567

Format Options:
A) Local
B) Domestic
C) International
D) Odd

Input Option:  C

Phone Number:  +1-555-123-4567
```

Example 4:

```
Input Phone Number:  5551234567

Format Options:
A) Local
B) Domestic
C) International
D) Odd

Input Option:  D

Phone Number:  0555 0123 4567
```

Example 5:

```
Input Phone Number:  5551234567

Format Options:
A) Local
B) Domestic
C) International
D) Odd

Input Option:  F

Error:  Invalid Option!
```

Example 6:

```
Input Phone Number:  Not a number

Error:  Invalid Phone Number!
```

Example 7:

```
Input Phone Number:  1 2 3

Error:  Invalid Phone Number!
```

Example 8:

```
Input Phone Number:  1234567

Error:  Invalid Phone Number!
```

Question 2 (23 Marks)

Write a C program named ***dates.c*** that determines which of the entered dates comes later on the calendar. The user may enter any number of dates. The user will enter 0/0/0 to indicate that no more dates will be entered.

Dates will be entered in the `mm/dd/yy` format where `mm` is a two digit month number (1 to 12), `dd` is a two digit day of the month (1 to 31) and `yy` is a two digit year (00 to 99). You should print an error if an invalid date is entered (does not follow the format or out of the given ranges) and ask the user for a new date (do not exit).

The user is allowed to input a one digit number (they do not need to pad it with a zero). For example `3/6/8`, `3/6/08`, `3/6/18`, `3/16/18`, and `03/06/08` are all valid dates. When outputting, the latest date, you should always pad the day, month and year to ensure they are two digits. For example, `3/6/8` should be output as `03/06/08`.

You do not have to check that the number of days are correct for a given month. For example, you can consider `2/31/18` to be a valid date (February has 28 days). However, the number of days must be in the range 1 to 31. The same date maybe input multiple times.

If the user does not input any dates before entering 0/0/0 an error should be displayed and the program should exit with a non-zero exit status. Otherwise the exit status should be zero.

Hint: *You do not need to use strings or arrays for this, nor do you need to store every date.*

Example Input/Output (input in green):

Example 1:

```
Enter a date (mm/dd/yy):  3/6/08
Enter a date (mm/dd/yy):  5/17/07
Enter a date (mm/dd/yy):  6/3/07
Enter a date (mm/dd/yy):  0/0/0
03/06/08 is the latest date
```

Example 2:

```
Enter a date (mm/dd/yy):  10/2/18
Enter a date (mm/dd/yy):  0/0/0
10/02/18 is the latest date
```

Example 3:

```
Enter a date (mm/dd/yy):  1/1/1
Enter a date (mm/dd/yy):  12/31/99
Enter a date (mm/dd/yy):  05/05/05
Enter a date (mm/dd/yy):  5/5/5
Enter a date (mm/dd/yy):  12/31/99
Enter a date (mm/dd/yy):  9/15/15
Enter a date (mm/dd/yy):  0/0/0
12/31/99 is the latest date
```

Example 4:

```
Enter a date (mm/dd/yy):  0/0/0
Error:  At least one date must be input!
```

Example 5:

```
Enter a date (mm/dd/yy):  7/0/8
Error:  Invalid day number!
Enter a date (mm/dd/yy):  0/7/8
Error:  Invalid month number!
Enter a date (mm/dd/yy):  7/7/-8
Error:  Invalid year number!
Enter a date (mm/dd/yy):  13/7/8
Error:  Invalid month number!
Enter a date (mm/dd/yy):  7/32/8
Error:  Invalid day number!
Enter a date (mm/dd/yy):  7/7/100
Error:  Invalid year number!
Enter a date (mm/dd/yy):  1/1/18
Enter a date (mm/dd/yy):  03/1/19
Enter a date (mm/dd/yy):  0/0/0
03/01/19 is the latest date
```

Example 5:

```
Enter a date (mm/dd/yy):  -10/7/17
Error:  Invalid month number!
Enter a date (mm/dd/yy):  10/-7/17
Error:  Invalid day number!
Enter a date (mm/dd/yy):  0/0/0
Error:  At least one date must be input!
```

Example 6:

```
Enter a date (mm/dd/yy):  potato
Error:  Invalid date format!
Enter a date (mm/dd/yy):  10/10/pizza
Error:  Invalid date format!
Enter a date (mm/dd/yy):  10/$7/9
Error:  Invalid date format!
Enter a date (mm/dd/yy):  10\7\9
Error:  Invalid date format!
Enter a date (mm/dd/yy):  2/31/18
Enter a date (mm/dd/yy):  4/31/17
Enter a date (mm/dd/yy):  0/0/0
02/31/18 is the latest date
```

Question 3 (31 Marks)

The area of a circle of radius r is given by:

$$Area\ of\ a\ circle = \pi \times r^2$$

Imagine that you divided this circle exactly into 4 quadrants. The area of one quadrant is then $0.25 \times \pi \times r^2$.

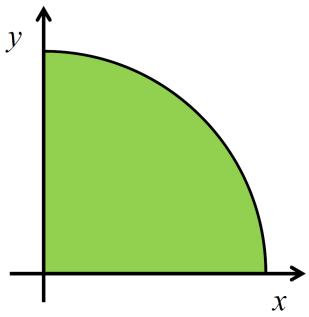
Let us set the radius of this circle to be $r = 1$. The equation hence becomes:

$$Area\ of\ one\ quadrant\ of\ a\ circle\ of\ radius\ 1 = 0.25 \times \pi$$

We can simply say that:

$$\pi = 4 \times the\ area\ of\ one\ quadrant\ of\ a\ circle\ of\ radius\ 1$$

To calculate the area of the shaded quadrant of this circle, you can use a random number generator and guess effectively the correct value of the constant π.



The steps of this technique are as follows:

1. Generate a random real number between 0 and 1 and assign it to x.
2. Generate a random real number between 0 and 1 and assign it to y.
3. If $x^2 + y^2 \leq 1$, it means that this (x, y) coordinate lies inside the shaded quadrant.
4. Repeat these steps N times, where N is ***sufficiently large number***.
5. Calculate the ratio of the points located inside the circle to the total number of generated points, i.e., N. This ratio should represent the ratio between the area of one quadrant of this circle to the area of 1×1 square. That is, this ratio approximates the area of one quadrant of this circle.
6. Multiply the calculated ratio by 4 to find an approximation of the mathematical constant π.

Use the above procedure to create a C program named ***pi.c*** to approximate the mathematical constant π. This program should contain a function named `estimate_pi` with the following function prototype:

```
double estimate_pi(long long n);
```

This function should take the value of N and return an estimation of pi using the technique described in this question.

Your program must read (from standard input) and validate the value of N (a positive whole number) from the user at the beginning of execution. You should print an error and exit with a non-zero exit status if your program fails to read in or validate the value of N, otherwise exit with an exit status of zero.

Write a loop inside your program to recalculate the value of π ***10 times*** using your `estimate_pi` function and the given value of N from the user. Print each estimated value of pi with 10 digits of precession and calculate the mean (the average value) and the standard deviation for results. Also print the mean and standard deviation with 10 digits of precession.

$$mean = \frac{\sum_{i=1}^{i=n} x_i}{n} \qquad standard\ deviation = \sqrt{\frac{\sum_{i=1}^{i=n} x_i^2}{n} - mean^2}$$

Test your program and ensure it works correctly for the following values of N (you do not need to provide this output with your submission):

$$N = 10,\ N = 100,\ N = 1{,}000,\ N = 100{,}000,\ N = 10{,}000{,}000$$

Question 4 (23 Marks)

Write a C program named ***change.c*** that determines the smallest number of $20, $10, $5, $2, and $1 bills/coins necessary to pay a dollar amount. The function prototype must be as follow:

```
void pay_amount(int dollars, int *twenties, int *tens, int *fives, int *toonies,
int *lonnies);
```

where the dollar amount is represented by the `dollars` parameter. The `twenties` parameter points to a variable in which the function will store the number of $20 bills required. The `tens`, `fives`, `toonies` and `lonnies` parameters are similar.

To test your function, write a `main` function that asks the user to enter a positive integer value (to be validated by your program). Consider this value to be the dollar amount (`dollars`) and get the smallest number of bills/coins necessary to pay this amount, and then display the values returned by the function.

If invalid input is entered (not a number or negative) you program should exit with an error message and non-zero exit status, otherwise return an exit status of zero.

Test your code and ensure it works correctly with the following values (you do not need to provide this output with your submission): 118, 81, 12, and 1.