ITSC 2181 – Introduction to Computer Systems
Spring 2024
Module 05 – Unit 1: Lab

Objectives

- Practice how to write small C programs, including identifying and correcting `C` compiler error messages and warnings.
- Practice the use of **pointers** in C.
- Practice the use of **dynamic memory allocation** in C.
- Practice the use of functions from the **string library** (`string.h`)

General Instructions

- Please do not write your email or user ID anywhere in the program. To identify your code, you may use your UNC Charlotte 800#

- In this lab, you will write a few short C programs. Each program needs to be in its own source code file, i.e., a file with the *`.c`* extension.

- **You need to test your code thoroughly before submitting it**.

- **To earn any credit, a program has to compile.** *You may comment-out lines that have errors to obtain partial credit for work done.*

- Programs need to compile cleanly to receive full credit, i.e., they do not produce any errors or warnings.

- It is essential that you **do your own work.**

    - **Do not use any external resources** (Internet, AI, friends, etc.). All cases of cheating will be taken very seriously.

    - **If you need help, please ask the instructor, TA/IA or CCI Tutoring center.**

Program 1: Bills Calculation (50 points)

Write a program named `bills_needed.c` that does the following:

1. Asks the user to enter a dollar amount.

2. Shows how to pay the amount using the smallest number of $20, $10, $5, and $1 bills. *Assume that the amount is a whole number, i.e., you do not need to deal with cents.*

3. The program **must** include a function with the following prototype:

   ```
   void calc_bills(int dollar_amount, int *twenties,
                   int *tens, int *fives, int *ones);
   ```

4. The program **must** use the `calc_bills()` function to figure out the number of bills needed.

5. The program **cannot** use any global variables.

Hints:
- Do not use floating point numbers.
- Divide the amount by 20, then reduce the total amount by the amount paid in $20 bills and repeat for the other bill denominations.

A few sample runs are provided below:

```
Enter dollar amount to pay: 93

You need:
   $20 dollar bills: 4
   $10 dollar bills: 1
    $5 dollar bills: 0
    $1 dollar bills: 3
```

```
Enter dollar amount to pay: 200

You need:
   $20 dollar bills: 10
   $10 dollar bills: 0
    $5 dollar bills: 0
    $1 dollar bills: 0


Enter dollar amount to pay: 157

You need:
   $20 dollar bills: 7
   $10 dollar bills: 1
    $5 dollar bills: 1
    $1 dollar bills: 2
```

Program 2: String Concatenation (50 points)

We have provided a program named `concat_test.c` (see *Canvas*). This program is incomplete.

You need to implement the code for the `concatenate()` function, as follows:

1. Your implementation must use **dynamic memory allocation** to create a new string that is large enough to hold the contents of both strings, plus an extra space and the standard null character (`\0`).

2. You **cannot** use arrays.

3. The `concatenate()` function concatenates (links together as one) the two strings that are passed in the input parameters and adds a space in between them. We recommend using the `strcpy()` and `strcat()` functions from the C Standard library in the implementation; however, you may write all the code from scratch if you want. See the following resources:

   https://cplusplus.com/reference/cstring/strcpy/

   https://cplusplus.com/reference/cstring/strcat/

4. The `concatenate()` function needs to work for strings of **<u>any</u>** length. Do not hard-code the lengths or assume a specific maximum length.

5. Below are a couple of sample runs of the program, once the `concatenate()` function has been implemented.

```
First string: The United States
Second string: of America

The two strings concatenated: The United States of America

First string: The University of North Carolina
Second string: at Charlotte

The two strings concatenated: The University of North
Carolina at Charlotte
```

Submission Instructions

1. Create a folder (directory) on your computer.

2. Name the folder **`ITSC_2181_M05_U1_Lab_`*`student-id`***
   Replace *student-id* with your UNCC student ID (800#), e.g., *8001231234*

3. Download and copy your program (source code) files into this folder.

4. You should consider keeping the files for every lab in a separate folder (directory) from your other course materials.

5. Submit the source code (`.c` files) of your programs via *Canvas*.

Grading Rubric

- This lab is worth a total of **100 points**.

- **Do your own work. Do not use any external resources** (Internet, friends, etc.). All cases of cheating will be taken very seriously. **If you need help, please ask the instructor, TA/IA or CCI Tutoring center.**

- Programs need to compile to earn any credit. *You may comment-out lines that have errors to obtain partial credit for work done.*

- Your work will be graded on three (3) major components: Logic and flow of program, output, and formatting/organization. Refer to the following table for details.

Logic and Flow of Program - 60%	
Fully Correct and code compiles without errors.	Full Credit
Minor Errors and code compiles without errors OR some required functionality is missing.	75% Credit
Major Errors and/or code compiles with warnings OR significant portions of the required functionality are missing.	50% Credit
Completely incorrect/missing/does not compile.	No Credit
Output - 30%	
Fully Correct and matches formatting and layout of sample output provided.	Full Credit
Minor Errors.	75% Credit
Major Errors.	50% Credit
Completely incorrect output.	No credit
Formatting/Organization of Code - 10%	
Code is clear, easy to read and formatter according to the guidelines. Whitespace has been used appropriately, including indentation and blank lines. Comments are used when needed.	Full Credit
Needs minor improvement.	75% Credit
Needs major improvement.	50% Credit
No formatting/organization at all.	No Credit

Additional Deductions

- Code that produces warnings: -10% per program
- Incorrectly named files: -2 points per file
- *Students who cheat on <u>any</u> course assignment, lab, test or other activity will have their course grade reduced by one letter grade, regardless of the activity's point value, if it is their first offense at UNC Charlotte.*