Lab 6

Introduction to C
CSCI 112, Spring 2015

Objectives

- Learn how to use 1D arrays
- Practice converting an algorithm to code
- Practice self guided program design

Overview

For this lab, you will write a program that prompts the user to enter the 12 digits of a UPC barcode, which you will store in an array. Using the algorithm described below, you will determine if the barcode is valid or invalid.

Algorithm

A barcode scanner for Universal Product Codes (UPC) verifies the 12-digit code scanned by comparing the code's last digit (called a *check digit*) to its own computation of the check digit from the first 11 digits. The process of validating a barcode is described below.

1. Calculate the sum of the digits in the odd-numbered positions (first, third, ..., eleventh). Multiply this sum by three.

   Protip: In C arrays, we count array indices starting at zero. Therefore, the "first" digit of the barcode with be stored in the 0th position of your array. Be careful to manage your array indices correctly.

2. Calculate the sum of the digits in the even-numbered positions (second, fourth, ..., tenth) of the barcode.

   Protip: Remember the 12th digit of the barcode is the checksum. Make sure not to include it in your sum of even indices.

3. Add these two sums together.

4. From this combined sum, extract the last digit. If the last digit is a 0, then 0 is the check digit. Otherwise, subtract the last digit from 10 to calculate the check digit.

PROTIP: To extract an individual digit from a number, use the modulus operator.

5. If the check digit matches the final digit of the 12-digit UPC, the UPC is valid. If not, the UPC is invalid.

Program

Write a program that prompts the user to enter the 12 digits of a barcode, separated by spaces. Your program will store the digits in an integer array, calculate the check digit, and compare your calculated check digit to the final barcode digit (the true check digit). If the digits match, the barcode is valid. Otherwise the barcode is invalid. You will print out your intermediate calculations for each Step 1 - 5. See the `Example Execution` section to determine how to format your results.

PROTIP: Practice good program design by splitting your program into subtasks and write a function to solve each subtask. Do not write all your code in your `main` function.

Example Execution

```
$ ./program6

Enter a bar code to check. Separate digits with a space >
0 7 9 4 0 0 8 0 4 5 0 1

You entered the code: 0 7 9 4 0 0 8 0 4 5 0 1
STEP 1: Sum of odds times 3 is 63
STEP 2: Sum of the even digits is 16
STEP 3: Total sum is 79
STEP 4: Calculated check digit is 1
STEP 5: Check digit and last digit match
Barcode is VALID.
```

Compile & Test

Compile your program using this `gcc` command. `c99` is a shortcut for running `gcc -std=c99`, which uses the C99 standard instead of the default C89 standard.

```
$ c99 lab6.c -o program6
```

PROTIP: Make sure you output what is expected, nothing more and nothing less.

Self Check

Once you finish, you can test out your program with the following barcodes.

```
0 7 9 4 0 0 8 0 4 5 0 1       Valid
0 1 1 1 1 0 8 5 6 8 0 7       Valid
0 5 1 0 0 0 1 3 8 1 0 1       Valid
0 2 4 0 0 0 1 6 2 8 6 0       Invalid
```

These barcodes are provided to allow you to verify that your program is working correctly. There is nothing you need to submit for this section.

Resources

- You should attend the Friday Lab session to seek assistance from the TAs and CAs.
- For general questions, check the D2L `Lab 6 Help Forum` to see if another student has already asked your question. Otherwise, post your question on the D2L forum. This forum is intended for general questions that will benefit all students. Do not paste your code nor give away any specific answers to the lab on this forum.
- For specific questions, attend the weekly lab session or attend the TA's office hours.
- You are encouraged to use resources or tutorials on the internet to learn unix or C. Check the class resource list on D2L for some links to useful resources.

Submission

- Assigned: March 17th
- Lab Day: March 20th
- Due Date: March 27th, 8:00am

1. Make sure you included ample and informative comments – it is 20% of your grade!
2. Rename your C file to `last_first_lab6.c` and substitute your last and first name.

   Protip: If you fail to follow the above file naming conventions, your program may not be graded automatically and you will lose points.

3. Submit your `.c` file to the D2L dropbox Lab6.

   Protip: Submit only your C file. Do not submit your object file or your executable program. Do not archive (e.g., `zip`) your file.

Each student will complete and submit this assignment individually. Submit your work before the deadline as late work will receive a 50% penalty. Labs submitted more than 24 hours late will not be accepted. The late deadline is Saturday, March 28th, 8:00am.