INTRODUCTION TO COMPUTER PROGRAMMING: FUNDAMENTALS OF C

Gilbert Pajela

Summer 2017

→ Lab 3: Functions

1. Write a program that prompts the user for three numbers then calls the function sort to display three numbers in decreasing order. You can use the following function prototype:

```
void sort(double num1, double num2, double num3);
```

2. Consider the standard decimal representation of a non-negative integer num. You wish to write a function that repetitively adds up the digits of a number until it eventually results in a single-digit number. For example, given the number 234567, the first iteration results in 27 ($= 2+3+4+5+6+7$), and the 2nd iteration results in 9 (=2+7). Since 9 is a single-digit number, no more iterations are needed and 9 is returned. Write a function with the following prototype to do this:

```
/* Precondition: num > 0 */
/* Postcondition: the return value is the iterated sum of
   digits of num */
int sumDigits(int num);
```

   To do this, we first need to identify functions that are useful to doing this. In this case, a list of useful function prototypes might be:

```
/* Return the index'th digit of num: */
int getDigit(int num, int index);
/* Return the number of digits in num: */
int numDigits(int num);
```

You should have at least these three functions, though you may have additional functions if you wish. Don't forget to write pre/post conditions for these functions. To test your program, write a main function that prompts for an input num, calls sumdigits and outputs the result. The program should also do error checking to ensure that inputs are legal, and reprompt as needed. (Note: For this problem, you are *not* allowed to convert numbers into strings and use string processing primitives.)

3. Consider again the standard decimal representation of a non-negative integer num, and additionally digits d and d'. You wish to write a function that replaces every d in num with d'. For example, if num=16762, d=6, d'=8, you wish to return 18782. The prototype of the function is:

```
// Precondition: num > 0
// Postcondition: ...
int replaceDig(int num, int oldDigit, int newDigit);
```

You should use `getDigit()` and `numDigits()` from the previous problem, though you may have additional functions if you wish. Don't forget to write pre/post conditions for these functions. To test your above program. write a main function that prompts for and inputs the 3 inputs, calls replaceDig, and outputs the result. The program should also do error checking to ensure that inputs are legal, and reprompt as needed. (Note: again, you may *not* convert numbers into strings and use string processing primitives.)

Exercises are adapted from the following sources:

1. Liang, Y. D. (2007). *Introduction to Java programming: comprehensive version.* Pearson Prentice Hall.

2. Shankar, S. (2017). CSCI 136 Lab Instructions.