

# CS116 - Intro to Programming, C++

Summer 1998

## Homework 1

Due in class Monday 8/3/98

### Before you start...

Before you get started on this assignment, add the following line to the end of your `.cshrc` file, which is located in your home directory (the directory you are in right after you log in):

```
setenv PATH /usr/local/classes/current/CS116/01/bin:$PATH
```

UNIX is case-sensitive, so type in the lower & upper case letters exactly as shown. Now save the file, log out, and log back in. This enables you to use the facility for turning in your homework into the electronic course dropbox (described below). You are now ready to proceed with the assignment.

### How to do this assignment

In your home directory, *create a new directory* for this assignment. This is the directory which will contain all your files, and you will do all of your editing/compiling/linking/testing here. Switch to this directory, and start writing your programs.

Make sure your programs are correct by testing them extensively, i.e. running them on many different inputs. If you can't get a program fully to work, make sure you know on what kind of input it works and on what kind of input it fails. This information will be very valuable for debugging.

Along with the correctness of your programs, you must pay attention to good *programming style*. For now, this means that 1) you use meaningful and descriptive variable and function names, and 2) you put plenty of comments in your code.

Once you are done with your programs, you will create a writeup file called **README**. In this file, you will

- Give a brief overview of the design of your programs.
- Explain which programs work correctly.
- Explain which programs do not work correctly (if any), what specifically are the problems, and what you think causes the problems.

You are now ready to turn in your homework. To turn in an online copy of your assignment, type the following (in your assignment directory):

```
> submit hw1
```

You will see some output generated by the `submit` program, while it makes a copy of your directory and places it in the dropbox.

In addition to the online submission, you will turn in a hardcopy of all your source files together with your **README** in class. You can print a file on the Maclab A laser printer via the command

```
> enscript -2r -PmaclabA filename
```

### Problem 1 [ 5 pts ]

Write a program called `min` which asks the user for two integers, and outputs the smaller one. This is what a sample run would look like this (here “>” represents the UNIX shell prompt, and user input is in italic):

```
> min
Please enter two integers:  13 6 <return>
The smaller integer is 6.
>
```

### Problem 2 [ 10 pts ]

Write a program called `sum` which asks the user for two integers such that the first is not larger than the second. It then computes and outputs the sum of all numbers between the two. For example, if the user enters 7 and 12, your program will output  $7 + 8 + 9 + 10 + 11 + 12 = 57$ , and if the user enters -2 and 3, your program will output  $(-2) + (-1) + 0 + 1 + 2 + 3 = 3$ . Your program must deal with bad input: if the first number is larger than the second, `sum` should output an appropriate error message and exit.

Here are a few sample runs:

```
> sum
Please enter two integers, the first not larger than the second:  7 12 <return>
The sum of all integers from 7 to 12 is 57.
> sum
Please enter two integers, the first not larger than the second:  -2 3 <return>
The sum of all integers from -2 to 3 is 3.
> sum
Please enter two integers, the first not larger than the second:  12 1 <return>
sum:  ** error:  first number larger than second.  Exiting.
>
```

### Problem 3 [ 10 pts ]

Write a program `count` which repeatedly prompts for and reads integers from the user, until a 0 is entered, at which point it will output a summary, reporting how many positive and how many negative numbers were read (not including the 0).

Here are two sample runs:

```
> count
Please enter an integer (0 to quit):  5 <return>
Please enter an integer (0 to quit):  -1 <return>
Please enter an integer (0 to quit):  -10 <return>
Please enter an integer (0 to quit):  8 <return>
Please enter an integer (0 to quit):  13 <return>
Please enter an integer (0 to quit):  0 <return>
Summary:
3 of the numbers were positive,
2 of the numbers were negative.
> count
Please enter an integer (0 to quit):  0 <return>
Summary:
0 of the numbers were positive,
```

```
0 of the numbers were negative.  
>
```

**Problem 4 [ 15 pts ]**

Write a program called `10questions` which plays the “10 Questions” game discussed in class. Your program will start out by asking the user to choose an integer between 1 and 1000. Then it will proceed with 10 questions of the form “Is your number greater than  $n$ ?”, where  $n$  is an integer determined by your program. To each question the user answers either ‘y’ or ‘n’. At the end of the 10 questions, your program should output the user’s number.

Here is a sample run:

```
> 10questions  
Please choose an integer between 1 and 1024...  
Is your number bigger than 512 ? (y/n) n <return>  
Is your number bigger than 256 ? (y/n) y <return>  
Is your number bigger than 384 ? (y/n) y <return>  
Is your number bigger than 448 ? (y/n) n <return>  
Is your number bigger than 416 ? (y/n) n <return>  
Is your number bigger than 400 ? (y/n) n <return>  
Is your number bigger than 392 ? (y/n) y <return>  
Is your number bigger than 396 ? (y/n) y <return>  
Is your number bigger than 398 ? (y/n) n <return>  
Is your number bigger than 397 ? (y/n) n <return>  
Your number is 397.  
>
```