CISC 181 sections 080-081 (Conrad) Midterm I March 15, 2004

Name____

Circle one:

Freshman Not-a-freshman
(I have to grade the freshman papers by Friday).

General Instructions

- DO NOT WRITE YOUR NAME ON ANY PAGE EXCEPT THIS ONE!
- You have 50 minutes
- **Pace Yourself!!!!!**

  Pay attention to the point values. When there are 10 minutes left, skim through and be sure you have at least written *something* for the questions that are worth many points.
- Read *all* the directions *carefully* on each problem.
- Good luck.

1. (30 pts) Write a compete C++ program to solve the following problem, including

   - an opening comment (*don't* put your name in the comment! -2 pts if you do!)
   - all necessary "stuff" that goes before the main program
   - a full main program complete with comments

   Problem Statement: Ask the user of the program to input three non-negative numbers. Determine which of the three numbers is largest. Then, scale the numbers down by dividing all three by the largest one. Then output all three numbers.

   Examples:

   - For input of 3 9 10, the output should be 0.3 0.9 1
   - For input of 45 180 90, the output should be 0.25 1 0.5

   Your program should provide appropriate prompts to the user, and should also label the output appropriately and neatly.

   There are two error conditions you should check for. If either of these error conditions occurs, print an error message, and terminate the program immediately.

   - If any of the numbers input is negative.
   - If the largest of the three numbers is zero. (Note that this can only occur if all three are zero.)

Extra space in case you need it

2. (10 pts) One of your lab exercises involved taking a number apart, digit by digit, by using division and the modulus operator. This questions tests your knowledge of that concept, plus your understanding of recursion.

   Write a C++ function (just the function, not a complete C++ program) with the following prototype:

```
int sumOfDigits(int x);
```

   The function must use recursion (not iteration) to return the sum of the digits in a number. Negative values are treated as if they were positive. Examples:

   - If you pass in 345, the function returns 12, since $3 + 4 + 5 = 12$
   - If you pass in 12, the function returns 3, since $1 + 2 = 3$
   - If you pass in -12, the function still returns 3 (treat negative numbers as if they were positive).
   - If you pass in 7, the function returns 7.

   Clues:

   - If the number passed in is negative, multiply it by -1 right away.
   - For any number between 0 and 9, the function returns the same number you passed in. This is your base case.
   - For the other case, consider the following:
     - get the final digit using the mod operator
     - add that to the result of a recursive call on the number you get when you strip off the final digit (you can do that by dividing by 10).

Extra space in case you need it

3. (10 pts) One of your lab exercises required you to write two different versions of a function that calculates the nth Fibonacci number, one using recursion, and the other using iteration. This question is a variation on that exercise.

   Write a function (only the function definition, not a complete C++ program) that will take one integer parameter called *x* and return an integer result.

   The function should be called `isFibonacci`. It should:

   - return 1 if the number x is a Fibonacci number
   - return 0 if x is *not* a Fibonacci number.

   For example, your function should return 1 if you pass in any of the following values: 0, 1, 2, 3, 5, 8, 13, 21, 34, 55, etc.

   Your function should return 0, if you pass in any negative number, or any non Fibonacci number such as 4, 6, 7, 9, 10, 11, 12, 14, etc.

   If is is helpful, you may assume that there is already a function available for your use with the prototype:

```
  int fibonacci(int n);
```

   that returns the nth Fibonacci number in the sequence. For example:

```
 fibonacci(0) returns 0
 fibonacci(1) returns 1
 fibonacci(2) returns 1
 fibonacci(3) returns 2
 fibonacci(4) returns 3
 fibonacci(5) returns 5
 fibonacci(6) returns 8
 etc...
```

   You can assume that this implementation of Fibonacci is efficient and correct, so if it makes things easier, you may use it in your answer (i.e. call it from inside your `isFibonacci` function.)

Extra space in case you need it

Editor Commands

Circle either vi or emacs, to indicate which editor you prefer. Your choice will determine the correct answer to each of the following questions.

vi   emacs

4. (1 pts) What do you type at the command line to invoke your editor to create a new file called myprog.cpp

5. (1 pts) What keystrokes do you use to save your file, and quit the editor?

6. (1 pts) Suppose you are editing a very large C++ program and you want to search for the function called `recursiveFactorial`. You want your editor to jump right to the first occurence of the name `recursiveFactorial`? What key do you press?

7. (2 pts) Describe the keystrokes needed to "cut" and "paste" (or alternatively, how to "copy" and "paste") in your editor.

   Note: DO NOT describe cut and paste functions of the ssh program, or of the X Windows environment, or the "menu" style commands in vim or emacs. I want the native keystrokes, as described in chapter 14 or 15 of your Anderson text.

Short Answer

8. (1 pts) What is the stream extraction operator?

9. (2 pts) Rewrite the following piece of code using a ternary operator.

```
if (x < 0 )
  absX = x * -1;
else
  absX = x;
```

10. In the late 1960s, Dijkstra wrote a famous letter criticizing a certain programming language feature. In his letter, he cited a result (a proof) published by Bohm and Jacopini.

    (a) (1 pts) What feature of the language did Dijksta criticize (just name it, no need for long explanation):

    (b) (3 pts) What result of Bohm and Jacopini did he cite? (explain *briefly*.)

11. Number conversions:

    (a) (4 pts) Convert 68 from decimal to binary

    (b) (4 pts) Convert 1F7C from hexadecimal to binary

12. (2 pts) Write an example of a preprocessor directive

13. Describe the functions of each of the following parts of a computer.

    (a) (2 pts) CPU

    (b) (2 pts) RAM

14. When calling a function, a one of the things that gets pushed on the stack as part of the call frame is the current value of the program counter (PC).

    (a) (2 pts) What is the program counter? (describe briefly)

    (b) (2 pts) Why it necessary to push its value on the stack (as part of the call frame) when calling a function?

15. Suppose I start with an empty stack, and then do the following sequence of operations:

- push 19
- push 26
- push 13
- push -4
- pop
- *pop*
- push 11
- push 16
- pop
- push 12

(a) (2 pts) What value will be returned as a result of the *second* pop operation (the one in *italics*?)

(b) (3 pts) Draw what the stack will look like at the end of the sequence of operations?

16. Consider the C++ program on the following page.

   (a) (6 pts) Give the output

   (b) Draw what the stack will look like at each of three points in time (you should end up with three drawings of stacks).

   - (3 pts) just before the call to function `helloThere` from `main`.
   - (3 pts) inside function `helloThere`, just *before* the `return (y + a * x)` is executed.
   - (3 pts) back in function `main`, just before the `return 0` is executed.

(Note that for this question, we not talking about just any old stack, we are talking about "the stack", the one where call frames are pushed and popped, and local automatic variables are stored.)

```cpp
// helloThere.cpp

#include <iostream>

using std::endl;
using std::cout;

int helloThere(int x, int y)
{
  int a = 5;

  cout << "Hello There!" << endl;

  return (y + a * x);

}

int main(void)
{
  int a = 2;
  int b = 3;

  int c = -4;

  c = a + 2 * b;

  cout << "c=" << c << endl;

  b = helloThere(a, c);

  cout << "a=" << a;
  cout << " b=" << b << endl;
  cout << " c=" << c << endl;

  return 0;

}
```

Total Points: 100