

NAME _____

SID (last 4) _____

LOGIN _____

CS 010 – Intro to Computer Science
Mid-term exam – Friday 4/23 – total 100 points

Spring - 2004
Version B

Time: 45 minutes

You may have on your desks ONLY this exam, a writing implement and an eraser.

You may use a separate sheet(s) of scratch paper. Make good use of this – don't write your answers in the assigned space until you are sure of them!

WRITE CLEARLY IN THE SPACE PROVIDED: **illegible responses will not be graded**

You will be required to show your student ID when you hand in your exam.

Be sure to read each problem carefully and follow the directions.

Problem 1	16	
Problem 2	16	
Problem 3	15	
Problem 4	18	
Problem 5	15	
Problem 6	20	
TOTAL	100	

NAME _____

SID (last 4) _____

LOGIN _____

1. (16 pts - 2 each) True/False. Please write out "True" and "False". Do not just write T or F.

TRUE

a. *NUMBER*, *sum_of_positives*, and *var2* are all examples of legal identifiers (variable, constant, or function names). TRUE

TRUE

b. The main difference between a do-while loop and a while loop is that the body of the while loop may never execute but the body of the do-while loop will always execute at least once.

TRUE

c. The following statement is a legal variable declaration statement.
`int number = 10, sum;`

TRUE

d. *const*, *int*, *char*, and *return* are all examples of keywords (reserved words) in C++.

TRUE

e. If at least one of the operands on either side of the division operator is a double, "double" division is performed rather than integer division.

TRUE

f. The variable *x* has the value 3 after the following statements.

```
int x;  
x = 3 % 5;
```

FALSE

g. If your program compiles and runs, but gives an incorrect output, this is known as a syntax error.

TRUE

h. Only one of the Boolean expressions on either side of an AND (&&) operator must be false for the entire expression to be false.

2. (16 pts - 4 each) Fill in the blank.

a. A compiler is a program that translates a high-level language program, such as a C++ program, to a machine-language program that the computer can understand.

b. For the function declared below, what is its return type? int

```
int my_func(char my_arg);
```

c. List one advantage of using functions. easier to understand, change, write, test, debug. code reuse, team development

d. The following cout statement has at least one syntax error.

```
cout << "She yelled, "Geronimo!", as she jumped out of the plane.\n";
```

Rewrite the statement so that it still outputs the following line but does not generate syntax errors:

She yelled, "Geronimo!", as she jumped out of the plane.

```
cout << "She yelled, \"Geronimo!\", as she jumped out of the plane";
```

NAME _____

SID (last 4) _____

LOGIN _____

3. (15 pts) Write an **if-else** statement that increments a variable called *positive_count* by one if the value of another variable called *number* is positive or 0. Otherwise the **if-else** should increment a variable called *negative_count* by one.

```
if (number >= 0)
{
    positive_count = positive_count + 1;
}
else
{
    negative_count = negative_count + 1;
}
```

4. (18 pts - 6 each) To the right of each of the following code fragments, show its output.

a.

```
int i = 20;
while (i > 10)
{
    cout << i << " ";
    i = i - 3;
}
cout << i << endl;
```

20 17 14 11 8

b.

```
int i = 20;
do
{
    i = i - 3;
    cout << i << " ";
} while (i < 10);
cout << i << endl;
```

17 17

c.

```
int x = 0;
while (x < 5)
{
    if (x < 2)
    {
        x = x + 2;
    }
    else
    {
        x = x + 1;
    }
    cout << x << endl;
}
```

2
3
4
5

NAME _____

SID (last 4) _____

LOGIN _____

5. (15 pts) Write a **function declaration** and **function definition** for a function that takes 2 doubles as arguments and returns the smallest of the 2 double arguments.

// declaration
double smallest (double arg1, double arg2);

// definition
double smallest (double arg1, double arg2)
{
 if (arg1 < arg2)
 {
 return arg1;
 }
 else
 {
 return arg2;
 }
}

NAME _____

SID (last 4) _____

LOGIN _____

6. (20 pts) Write a **program** that gets one integer at a time from the user until the user enters a negative number. When the user enters the negative number, the program should output the sum of all the numbers entered, except the negative number entered at the end. You are **NOT** required to use functions for this problem.

Sample run: (user input in **bold**)

```
Enter an integer: 5
Enter an integer: 4
Enter an integer: 10
Enter an integer: 20
Enter an integer: 1
Enter an integer: 3
Enter an integer: -20
```

The sum of all the positive numbers entered is 43.

```
#include <iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int sum = 0, number = 0;
```

```
    do
```

```
    {
```

```
        sum = sum + number;
```

```
        cout << "\n Enter an integer: ";
```

```
        cin >> number;
```

```
    } while (number >= 0);
```

```
    cout << "The sum of all the positive numbers entered is "
```

```
        << sum << "." << endl;
```

```
    return 0;
```

```
}
```