CS 180 Exam 2 October 17, 2018

Name: _____

Part 1: Relational and Logical operators (15 points)

Indicate the value of each expression:

3 < 0 ? "q" : "z"

"abc" < "abcd"

3 < 7 || 5 != 5

Show a Boolean expression that expression that expresses each of the following (assuming x and y are ints):

x is less than 7 and greater than or equal to 2

x does not equal 5 or y is less than 10

Part 2: Data Validation Loop (15 points)

Write code for a data validation loop that reads a value for the variable student_count until the value is in the range MIN_COUNT to MAX_COUNT inclusive.

```
const unsigned int MIN_COUNT = 0;
const unsigned int MAX_COUNT = 30;
int student_count;
```

Part 3: Accumulating from file data (20 points)

You are given a data file named "data.txt" that contains precisely 100 non-negative integers. Write a fragment of C++ code that declares necessary variables and constants, reads in the numbers. While reading the numbers calculate the minimum value, and the sum value. Print the minimum value and the average to the screen after all of the numbers have been read.

Part 4: Functions (15 points)

Write first the prototype, then the actual definition of a C++ function called fahrenheit_to_celsius that takes a Fahrenheit temperature (double) as input and returns the equivalent Celsius temperature (double). Be sure that your prototype is documented the way our style guide prefers. The formula for conversion (as defined by a math teacher) is:

$$C = 5/9(F - 32)$$

Part 5: Loop equivalence (15 points)

Given the following while loop:

```
const unsigned_int MAX_COUNT = 10;
unsigned int count = 0;
unsigned int total = 0;

while (count < MAX_COUNT)
{
  total += count;
  count++;
}

cout << total << endl;
```

Show a for loop that does the equivalent calculation:

Now show a do while loop that also does the equivalent calculation:

Part 6: Designing & Writing Code (20 points)

Some programming languages provide a division_result function that calculates both the integer quotient and the remainder when given two integers to divide. In C++ we can write a version of this function with four parameters. The first two parameters are pass-by-value and can be called dividend and divisor. The last two parameters are pass-by-reference and can be called quotient and remainder. The division_result function does not return anything.

Write the prototype and definition for a well-written C++ function that does this. As an example, if dividend is 13 and divisor is 5, then the value assigned to quotient should be 2 and the value assigned to remainder should be 3.