Assignment 1

PROBLEM SET 1

Standard namespace:

Recall from lecture that after you tell the preprocessor to use functions from the standard input/output library (#include <iostream>), you need to tell the compiler that the functions you'll be using are defined in the "standard namespace" (a language feature we won't go into in detail), which you do with "using namespace std;". Another way to access the functions in the standard namespace is to prefix each one with the name of the namespace. For instance, you can skip the using statement if you write std::cout and std::cin every time you want to print or input text.

Problem 1

Look at the following program and try to guess what it does without running it.

```
#include <iostream>
int main()
{
int x = 5;
int y = 7;
cout << "\n";
cout << x + y << " " << x * y;
cout << "\n";
return 0;
}
```

Problem 2

Put the following program in a file called buggy.cpp and compile it. (".cpp" is the standard suffix for C++ code files.) What errors do you receive? How would you fix them? (You may need to correct and recompile several times before all the problems are fixed.)

```
#include <iostream>
int main()
{
cout < "Hello World\n;
return 0}
```

Problem 3

Write a program that writes "I love C++" to the screen.

char * data type:

One data type that we discussed only implicitly in lecture is strings. Strings are just sequences of characters, which are denoted by quotation marks – e.g. "Hello world!" is a string, as are "\n", "The Noble Duke of York", and so on. (Note that strings use double quotes, whereas lone characters, such as '\n', use single quotes.) The official C++ name for this data type – i.e. the equivalent for strings of int, double, etc. – is char *. You do not need to know how this syntax works; all you need to know is that when you want to declare a variable to store a string, you type something like: char *loveCpp = "I love C++!";. (The * is usually placed directly before the variable name, but the name of this variable is just loveCpp.) The escape codes we talked about in lecture are all for encoding special characters as chars or within char *'s.

Problem 4

What would be the correct variable type in which to store the following information?

a. Your age

b. The area of your backyard

c. The number of stars in the galaxy

d. The average rainfall for the month of January

e. Your name

f. A status value corresponding to failure or success

Problem 5

Suggest a good variable name for each piece of information you described in Problem 4, and provide a sample declaration/assignment statement for the variable.

Expressions: An expression in C++ is any series of tokens that, when it is evaluated and all commands contained within it are run, is equivalent to a value.

Expressions that you type directly into your code – integers, decimal numbers, true, false, etc. – are ***constants. 35 and "This is a string" are constants, but 24 / 3 and myVariableare not, since they are calculated at run-time.***

There are two different types of expressions: ***L-values and R-values. An L-value appears on the left side of an assignment statement ("L" for "left"); it is the thing that is assigned to. An R-value is the expression whose value is evaluated, and possibly assigned to an L-value. An identifier – a name that you gave to something, such the name of a variable – can be used as an expression in either of these ways. For example, in the statement x = y + 5;, x and y are both identifiers; x is an L-value; and y, 5, and y+5 are all R-values.***

Problem 6

State whether each of the following is an expression or a statement. If it is an expression, state its data type, what type of expression it is (if it is a constant or an identifier), and the value it evaluates to. (Decimal numbers are doubles by default. You need not consider assignment statements to be expressions.)

a) 23.5
b) double a = 23.5;
c) 24 * 3.2
d) 24 / 32
e) 24 / 32.0
f) return 0;
g) x // Assume the line before says int x = 4;
h) 'x'

Problem 7

Write a program that prompts the user to enter two integer values and a float value, and then prints out the three numbers that are entered with a suitable message.

Problem 8

Write a program to evaluate the fuel consumption of a car. Have the user input the miles on the car's odometer at the start and end of the journey, and also the fuel level in the tank (in gallons) at the start and end of the journey. Calculate fuel used, miles travelled, and the overall fuel consumption in miles travelled per gallon of fuel. Print these values, accompanied by appropriate messages indicating what they are.

Named constants:

We talked in lecture about naming conventions for constants: a name in all-caps usually indicates a constant. C++ allows you to instruct the compiler to insist that a variable stay constant throughout the program. You can do this by putting the keyword const before the variable declaration. For example, const double PI = 3.14159; declares a variable named PI that can never be changed after its initial assignment to 3.14159. To attempt to change the value of such a variable – a *named constant – is a syntax error. Effectively, we've simply defined another name for the constant value 3.14159, which of course we aren't allowed to change.*

Problem 9

Write a program that sets the constant integer NUMBER_OF_VARIABLES to 2 and the integer variable x to 100. The program should then increment x, set it to the remainder of its current value divided by NUMBER_OF_VARIABLES, and add 2 to it. Use the operator shorthands as much as you can.