**CS173: Intermediate Computer Science**

**Reading 3**

Name: \_\_\_\_\_\_\_\_\_\_\_\_Daniel Lee\_\_\_\_\_\_\_\_\_\_\_\_\_

Read the assigned pages below from our course textbook. Complete the responses to the questions in this document and then save as a docx or pdf file. Submit your work by the assigned deadline on the Canvas course page or in class. Responses may be neatly handwritten or typed. **Put your name at the top!**

Readings: From the course textbook please read Chapter 3.

You should come away with understanding:

* the primitive C++ data types
* simple mathematical operators on numeric types
* how to use string datatypes

**1) Look at the chart in Section 3.1 What additional datatypes are listed here that we did not reference yet in class in our list of primitive datatypes?**

enum, long double

**2) How do we write Avogadro’s Constant as a C++ literal value? If you do not know Avogadro’s constant by heart, it’s available through the** [**Encyclopedia Britannica**](https://www.britannica.com/science/Avogadros-number)**. It is an important value in Chemistry and Physics.**

6.02214076E23

**3) Read the Software Engineering Tip on Page 99. Why is it a good idea to use named constants in place of numeric literals?**

It is a good idea to use named constants in place of numeric literals because it makes the program more readable, easier to modify, and ensures reliability.

**4) What is the difference between the unary – operator and the binary – operator? (The two minus operators).**

The difference between the unary - operator and the binary – operator is that unary – operator has just one operand and it is used for indicating the operand is negated while the binary – operator has two operands, and it is used for subtraction between two operands.

**5) What is the difference between the two groups of statements below? If you are unsure, put them both in a program and print the result of x and y.**

int x = 3; int x = 3;

int y = x++ + 3; int y = ++x + 3;

The difference between the two groups is that the group that has x++ has 6 for the value of variable y, which indicates that x++ increments the value of variable x after processing the calculation while the group that has ++x has 7 for the value of variable y, which indicates that ++x increments the value of variable x before processing the calculation.

**6) List the order of precedence table for the arithmetic operators:**

Highest precedence level: Unary + Unary -

Middle level: \* / %

Lowest level: + -

**7) What is the difference between type coercion and type casting? Give an example C++ statement of each kind.**

Type coercion is an implicit (automatic) conversion of a value from one data type to another (e.g. If there is a variable *x* declared as *float x;* and a statement *x = 16;* the compiler inserts extra instructions to convert 16 into 16.0 before storing it into *x*).

On the other hand, type casting is an explicit conversion of a value from one data type to another (e.g. The statement *int someInt = int(3.14);* will make the variable *someInt* to have the value of 3).

**8) Explain what each of the following string operations does.**

string word = “Happy”;

1. cout << word.length() << endl;

The length function, when applied to a string variable, returns an unsigned integer value that equals the number of characters currently in the string. The above statement would print out 5 which is the length of the variable word.

1. cout << word.find(“p”) << endl;

The find function searches a string to find the first occurrence of a particular substring and returns an unsigned integer value giving the result of the search. The above statement would print out 2.

word.find(“x”)?

cout << string::npos << endl;

1. cout << word.substr(3,2) << endl;

The substr function returns a particular substring of a string. The first argument is an unsigned integer that specifies a position within the string, and the second is an unsigned integer that specifies the length of the desired substring. The above statement would print out py.

(d) cout << word.at(0) << endl;

The at function allows access to characters directly

by their position. The above statement would print out H.