CS 121 – Week 5 Worksheet – Bunch of Programming Stuff

1. Using the below code and output, explain how default parameters work.

PROGRAM	OUTPUT

`#include <iostream>`
`#include <string>`
`using namespace std;`
`void printStuff(string str = "NO MESSAGE GIVEN")`
`{`
`  cout << str << endl;`
`}`
 	
`int main()`
`{`
`  printStuff("Hello!");`
`  printStuff();`
`  printStuff("Supercalifragilisticexpialidocious");`
`  printStuff("NO MESSAGE GIVEN");`
`  printStuff();`
`  printStuff("AND HIS NAME IS JOHN CENA\a");`
 	
`  return 0;`
`}`
	`Hello!`
	`NO MESSAGE GIVEN`
	`Supercalifragilisticexpialidocious`
	`NO MESSAGE GIVEN`
	`NO MESSAGE GIVEN`
	`AND HIS NAME IS JOHN CENA`
	
	`Default works like it sounds: If the user does not supply an argument to pass through, the default value is used in the function.`

2. Explain the difference between a literal in C++ and a constant in C++.

   A literal in C++ is the value portion of an expression (e.g. 'A', 7, “hello”) whereas a constant is a variable that does not change its value (e.g. *const int num*);

3. Look at the following function prototypes. Reading from top to bottom, note and explain the ones that are considered syntactically invalid in C++.

```
int addNums(int a, int b);                 // valid
double addNums(double a = 2.0, double b); // invalid (default starts from right)
float addNums(float a, float b = 3.0);     // valid (proper overloading)
short addNums(short a = 3, short b = 6);  // valid
long addNums(int z, int c = 3);            // invalid (overload issue with first
                                           //  prototype as both have same param)
```

4. What is the output of the following code segment?

PROGRAM	OUTPUT

`int x = 3;`
`int& y = x;`
`int z = y;`
`cout << x << " " << y << " " << z << endl;`
`y = 7;`
`cout << x << " " << y << " " << z << endl;`
	`3 3 3`
	`7 7 3`

5. Create a function that swaps two strings

```
swap(string& first, string& second) {
  string temp = first;
  first = second;
  second = temp;
}
```

6. Note two reasons where by-reference would be considered useful. (**HINT**: Think about how by-value works, the const keyword, and the size of some datatypes).

   When you want a function to return more than one value (e.g. function called getSinAndCos).

   When you want to send a large datatype into a function (since by-value copies the datatype, use *const* and by-reference to send it through). Example of a large datatype would be a vector of 10000 vectors of 1000 doubles, or later on classes and structures (which deal with OOP).

7. If you wanted to convert 26, a base-10 integer, to its base-2 (binary) representation, you would do something similar to the following:

$$
\begin{aligned}
26 / 2 &= 13 \text{ R } 0 \\
13 / 2 &= 6 \text{ R } 1 \\
6 / 2 &= 3 \text{ R } 0 \\
3 / 2 &= 1 \text{ R } 1 \\
1 / 2 &= 0 \text{ R } 1
\end{aligned}
$$

   Which leads to: $26_{10} = 11010_2$ (you read the bits from bottom to top).

   Using the above algorithm and the following incomplete code, create a function that takes in a base-10 integer and converts it to its binary representation.

```
string convBaseTenToTwo(int num)
{
  string bit_string = "";
  int quotient = num,
      remainder = 0;

  while(quotient != 0)
  {
    remainder = quotient % 2;
    quotient = quotient / 2;

    // insert to front of bit_string
    bit_string.insert(0, to_string(remainder));
  }

  return bit_string;
}
```