CS 121 – Week 4 Worksheet – Solution

1. What is the output of the following code? (The code below is on the website, “ws4-main.cpp"

OUTPUT:

5 8 13
12 144 5
b y
120 8 13
12 144 5
B B

Steps (i.e. effects of each function call):

FIRST SET OF FUNCTION CALLS:

1. int_three becomes what is returned by calling getSum(int_one, int_two). That is int_one plus int_two, which is 13.

2. dbl_two becomes what is returned by calling squareOf(dbl_one). That is dbl_one times dbl_one.

3. After the function call setDoubleToFive(dbl_three), dbl_three equals 5 + 5 – 5, which is 5.

FIRST SET OF OUTPUTS:
5 8 13
12 144 5
b y

SECOND SET OF FUNCTION CALLS:

4. int_one becomes the value returned by the call to factorialOf(5). Luckily, I defined the factorial correctly, so int_one becomes 5!, which is 120.

5. This was the trickiest. char_two becomes the value returned by the call to getUpperCase(char_one). The function checks if the ASCII value of ch (which is 'b' here) is greater than 90. As 'b' is equal to 98 and 98 > 90, subtracting ' ' (or 32 in ASCII) makes 'b' equal 66 (or 'B'), and returns 'B'. Note that if ch was less than 90, ch would be set to 'A' and return 'A'.

SECOND SET OF OUTPUTS:

120 8 13
12 144 5
B B

2. Define a function that swaps two double variables and show an example of how to call your function with two doubles called “hello” and “world”.

DEFINTION:

```
void swapDoubles(double& first, double& second)
{
        double temp = first;
        first = second;
        second = temp;
}
```

FUNCTION CALL:

```
swapDoubles(hello, world);
```

3. Assume the following is written in a C++ program and are prototypes (with definitions) but not compiled yet. Will the compiler throw errors, or will the below compile fine?

The below code does not compile.

This question was all about function overloading. All lines work except for the second one (since overloaded functions cannot have the same types and amount of parameters).

```
int subtract(int a, int b, int c);
double subtract(int d, int e, int f);       // this one prevents compilation

int subtract(int a, int b, int c, int d);
double subtract(int e, int f, int g);

int subtract(int a, int b, int c, int d, int e, int f, int g, int h);
float multiply(int a, int b = 4);           // "int b = 4" is called a default parameter
```