Practical 2 (Week 3)

The learning outcomes of this practical include:

- Learn to write a function in C++
- Understand and learn to use function overloading and default arguments.
- Understand the methods of parameter passing.

Task 2.1:

Redo **Task** 1.3 in the way that c) and d) are implemented in separate functions.

Task 2.2:

Given the following function definitions:

```
void testOverloading( int numerator, int denominator) {
     int fraction = numerator / denominator;
     cout << "Fraction1 = " << fraction << endl;
}

void testOverloading(double numerator, double
denominator) {
     double fraction = numerator / denominator;
     cout <<  "Fraction2 = " << fraction << endl;
}

void testOverloading(int numerator, double denominator) {
     double fraction = numerator / denominator;
     cout <<  "Fraction3 = " << fraction << endl;
}
```

write a main function with three function calls to each of the functions. Explain the outcome to your tutor.

Hint: An example of a function call to the third function:

```
     testOverloading(3,7.0);
```

Task 2.3:

Given the following function definition:

```
void testDefaultArg(int day = 1, string month= "Jan",
                    int year = 2017) {
     cout << "Today is " << day <<" "
     << month << " " << year << endl;
}
```

write a main function with the following function calls. Explain the outcome to your tutor.

```
testDefaultArg();
testDefaultArg(10);
testDefaultArg(10, "Jul");
testDefaultArg(10, "Jul", 2018);
```

Task 2.4:

Write a function that takes two integers, *numerator* and *denominator*, as input parameters and output their *quotient* and *remainder*. Use a main function test the function.

Hint: Use call by reference to output the quotient and remainder as parameters (see an example for Code for Lecture 2 (*TwoDigits.cpp*). Use the following formula to calculate quotient and remainder:

quotient = numerator / denominator;
remainder = numerator % denominator;

Task 2.5:

Write a function named *sort* that takes three integer parameters by reference. The function should rearrange the parameter values so that the first parameter gets set to the smallest value, the second parameter gets set to the second smallest value, and the third parameter gets set to the largest value. For example, given the variable assignments *a=30; b=10; c=20;* then the function call *sort(a,b,c)* should result in *a 10, b=20,* and *c=30*. Write also a main function to take input of three integers, call the function with the input and output the results of the function.