UNIT 03: C++ FUNCTIONS

Lesson 3

- Function Overloading
- Function Templates
- Random Numbers

Activity Requirements

- Most of the activities will involve writing programs. You will collaborate in groups of 2 to 3 students, but each of you will write your own programs on your own laptop.
- You will be submitting your source code at the end of the lesson for marks.
- You must be present, in class, to be eligible to participate in the activities, and thus be eligible for these marks.

Activity 7A

- In this activity, we will write two functions. The first will return the absolute value of an *int* argument. The second will return the abs value of a *double* argument. Both functions will be called ***AbsoluteValue***.
  1. Open the starter file *CPRG03-07A.cpp*
  2. Prototype and define the two *abs* functions based on the documentation provided in the comments.
  3. Run the program to test the functions.

CPRG03-07A.cpp

Activity 7A (cont'd)

- Defining two or more functions using the same name is called _____.
- Every function in the same scope must have a unique *signature*, which is the combination of the function name and _____.
- Good programmer code functions that have the same name to do similar actions. Technically, this _*<is/is not>*_ a requirement.

Activity 7B

- Save a copy of your Activity 7A code as ***CPRG03-07B.cpp***
  1. Remove the *AbsoluteValue(double)* function.
  2. Modify *AbsoluteValue(int)* so that it is a function template.
  3. Run the program to test the template.

Activity 8A

- Guided Activity:
  - We will modify a simple program that illustrates the use of *rand()* and *srand()*.
  - Open the starter file *CPRG03-08A.cpp*
  - Follow along with the instructor, modifying the code.

CPRG03-08A.cpp

Activity 8A (cont'd)

- The *rand()* function returns a _____ int value, between 0 and _____ .
- To initialize (seed) the pseudo-random number algorithm, use the _____ function.
- Seeding multiple times <u><*makes/does not make*></u> the values returned from *rand()* more random.

Activity 8B

- Save a copy of your Activity 8A code as ***CPRG03-08B.cpp***
  1. Create two more named constant s: *MIN_VALUE = 2* and *MAX_VALUE = 12*.
  2. Modify the assignment statement in the loop so that the random value is scaled between *MIN_VALUE* and *MAX_VALUE* (inclusively).
  3. Run the program to test your assignment statement.
  4. Change the *MIN_VALUE* and *MAX_VALUE* constants and re-run the program a few times to test your assignment statement.

Activity 8C

- Save a copy of your Activity 8B code as ***CPRG03-08C.cpp***
  1. Create an inline function called *ScaledRandomNumber*. This function should take two arguments representing the minimum and maximum random number desired. It should then return a random number in that range.
  2. Modify the assignment statement in the loop so that the random value determined by your function.
  3. Run the program to test your function.

Submission

- Open the *Unit 3- Lesson 3 Activities* Dropbox
- Attach your source code files *individually* (**DO NOT ZIP**) and submit
  - *CPRG01-07A.cpp*
  - *CPRG01-07B.cpp*
  - *CPRG01-08A.cpp*
  - *CPRG01-08B.cpp*
  - *CPRG01-08C.cpp*

Summary

- **Function overloading**: using same function name for more than one function
  - Compiler must be able to determine which function to use based on matching data types of pararmeter(s) and argument(s)
- Each function must be written separately
- Use of same name does not require code to be similar
  - Good programming practice: functions with the same name perform similar operations

Summary (cont'd)

- Where more than one overloaded function would only differ by the data-type(s) handled, a ***function template*** may be used instead.
  - One or more general data types are designated (i.e. T)
  - These general data types are replaced by actual data types when compiler encounters a function call

Summary (cont'd)

- **Random numbers**
  - Series of numbers whose order can't be predicted
  - In practice, finding truly random numbers is hard
- **Pseudo-random numbers**
  - Random enough for the type of applications being programmed
- All C++ compilers provide two general-purpose functions for generating random numbers
  - `rand()` and `srand()`

Summary (cont'd)

- **Scaling**
  - Procedure for adjusting the random numbers produced by a random-number generator to fall in a specified range
  - Scaling a random number as an integer value between *MINIMUM* and *MAXIMUM*

```
MINIMUM + rand() % (MAXIMUM – MINIMUM + 1)
```

References

Bronson, G. (2012). Chapter 6 Modularity Using Functions. In *A First Book of C++* (4th ed.). Boston, MA: Course Technology.