UNIT 03: C++ FUNCTIONS

Lesson 2

- Variable Scope
- Variable Storage Class
- Introducing the ctime header

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 4A

- Guided Activity:
  - We will write a simple program that illustrates the difference between local and global variables.
  - Open the starter file *CPRG03-04A.cpp*
  - Follow along with the instructor, modifying the code.

CPRG03-04A.cpp

Activity 4B

- Guided Activity:
  - We will write a simple program that illustrates the use of the scope resolution operator.
  - Open the starter file *CPRG03-04B.cpp*
  - Follow along with the instructor, modifying the code.

CPRG03-04B.cpp

Activity 4C

- Guided Activity:
  - We will modify a simple program that illustrates the ***extern*** and ***static*** storage classes for global variables.
  - Open the starter file *CPRG03-04C.cpp*
  - Follow along with the instructor, modifying the code.

CPRG03-04C.cpp

Activity 4D

The potential problems with using global variables outweighs the convenience of using global variables.

- Agree or Disagree? **Explain.**
  - Post to the Unit 3, Activity 4D topic in the forum.
  - One post per group or 2-3 students.
  - Point form is fine.

Activity 4D (cont'd)

- In-class Discussion:

The potential problems with using global variables outweighs the convenience of using global variables.

Activity 5A

- Write a simple test function that initializes a local variable to 1 and outputs it to the console.
- The final line of the function should increment the local variable by one.
  1. Open the starter file *CPRG03-05A.cpp*
  2. Code the body of the *Test* function based on the documentation provided in the comments.
  3. BEFORE you run the program, predict what the output should be.
  4. Run the program. Consider any difference between what you predicted and the actual output.
  5. Add the keyword ***auto*** before the data type of the variable declaration and re-run the program.

CPRG03-05A.cpp

Activity 5A (cont'd)

- By default, the storage class of local variables is _____.
- Functions _____ retain the value of ***auto*** local variables between function calls.
- The lifetime of an ***auto*** local variable is the lifetime of the _____.

Activity 5B

- Save a copy of your Activity 5A code as ***CPRG03-05B.cpp***
  1. Replace the keyword ***auto*** with the keyword ***static***.
  2. BEFORE you run the program, predict what the output should be.
  3. Run the program. Consider any difference between what you predicted and the actual output.

Activity 5B (cont'd)

- The ***static*** storage class allows a function to _____ local variable values between function calls.
- The lifetime of a ***static*** local variable is the lifetime of the _____.

Activity 6

- Guided Activity:
  - We will modify a simple program that illustrates the difference between ***auto*** and ***register*** local variables, using functions from the ***ctime*** header.
  - Open the starter file *CPRG03-06.cpp*
  - Follow along with the instructor, modifying the code.

CPRG03-06.cpp

Activity 6 (cont'd)

- Using the ***register*** keyword is a *request* that the memory for the variable is allocated from _____, instead of the computer's RAM.
- When the ***register*** storage class is not supported or cannot be implemented, variables declared with the ***register*** keyword are treated as _____.

Activity 6 (cont'd)

- ***time_t*** is a data type that, according to the current C++ standard, is the same as the data type _____, however this could change in future standards.
- ***time()*** is a function that returns the number of seconds that have passed since midnight, _____, GMT.
- ***ctime()*** is a function that converts a _____ _____into a _____.

Submission

- Open the *Unit 3- Lesson 2 Activities* Dropbox
- Attach your source code files *individually* (**DO NOT ZIP**) and submit
  - *CPRG01-04A.cpp, CPRG01-04B.cpp, CPRG01-04C.cpp*
  - *CPRG01-05A.cpp, CPRG01-05B.cpp*
  - *CPRG01-06.cpp*

Summary

- *Scope* determines where in the program the variable can be used.
  - A *local* variable can be used only in its defining function or block.
  - A *global* variable is defined outside a function and can be used in any function following the variable's definition.
    - Global variables are set to 0 when not initialized explicitly.
    - Global variables not declared as static can be shared between files by using the keyword *extern*.

Summary (cont'd)

- *Storage category* determines how long the value in the variable is retained (*lifetime*).
  - *auto* variables are local variables that exist only while their defining function is executing
  - *register* variables are similar to auto variables but are stored in a computer's registers rather than in memory
  - *static* variables can be global or local and retain their values while the program is running.
    - Static variables are set to 0 when not initialized explicitly.

References

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