**WEEK 8B: USER DEFINED TYPES – ENUM**

# Objectives

Following the completion of this topic, students should be able to:

* Understand the basic concepts of the user defined data type enum

**TASK 1. ENUMERATED TYPES**

An enumerated type is a user-defined type whose values are defined by a list of named constants of type int. When defining an enumerated type, you can use any int values and can define any number of constants. For example, the following enumerated type defines a constant for the speed limit of each area.

enum speedLimit {school = 40, residential = 50, builtUp = 60, highway = 100};

* If you do not specify any numeric values, the identifiers {school, residential, builtUp, highway} in an enumerated type definition are assigned sequential values beginning with 0.
* Definitions for enumerated types are *usually* placed in the global area so that they are accessible by all functions. Once an enum type has been defined, you can then declare enum variables. *Note: Do not confuse a definition**of an enum type with a declaration of an enum variable*.

The following statement,

speedLimit speed;

declares an enumerated variable of type speedLimit. Once declared, values can be assigned to the variable. The following statement, speed = school;

assigns the value of school to the variable speed. Thus, the following statement cout << speed << endl; will produce output of

40

In a new program, called *’task1.cpp’*:

1. Define an enum type named direction with the values NORTH, SOUTH, EAST and WEST. ***Note: Remember to compile the program regularly during construction.***
2. Declare a variable named dir of type direction.
3. Assign NORTH to the variable dir.
4. Output the value of the variable dir.
5. Advance dir to the next value in the list. Remember that arithmetic is not possible on enumerated types, so casting will be required.
6. Output the value of the variable dir.
7. Create a function which outputs the strings “NORTH”, “SOUTH”, “EAST” and “WEST” instead of the numeric values when passed a direction variable. Use a switch statement to accomplish this and **not** an if statement. The prototype is void printDirection (direction dir);
8. Using a for loop, call printDirection four times. Declare a new direction variable and use this to control the loop and pass to the function.
9. Run the program. What is the output?

The program prints North, South, East, West

1. Modify the for loop to iterate 5 times. What happened?

The program prints North, South, East, West, West