**LECTURE 4: DATA TYPES, VARIABLES AND OPERATORS**

* 1. **Introduction**

****So far we have discussed basic concepts in programming and in this lecture we will learn about data types, variables and operators. When writing a program, data types are used for modeling variables. That is, variables cannot be declared without the use of data type. Another concept that comes in handy in programming is operators. These are special characters that are used to connect code segments. Therefore, data types, variable declaration and use of operators will be discussed in this lecture.

The lecture is organized as follows;

1. Lecture objectives
2. Data type
3. Variables
4. Operators
5. Self-test questions
6. Summary
7. Reference
   1. **Lecture Objectives**

****At the end of this lecture the student should be able to;

1. Define explain various data types
2. Declare variable applying correct variable names in C
3. Declare and apply constants in C
4. Apply various operators in a C program
   1. **Data types**

** Data Types**

The C language provides standard means of modeling data known as datum. C supports five basic data types. The table below shows the five types, along with the C keywords that represent them. Don’t be confused by *void*. This is a special purpose data type used to explicitly declare functions that return no value.

|  |  |  |
| --- | --- | --- |
| **Type** | Meaning | **Keyword** |
| Character | Character data | char |
| Integer | Signed whole number | int |
| Float | floating-point numbers | float |
| Double | double precision floating-point numbers | double |
| String | String of characters | String |
| Void | Valueless | void |

The ‘int’ specifier

It is a type specifier used to declare integer variables. For example, to declare count as an integer you would write:

int length, width;

Integer variables may hold signed whole numbers (numbers with no fractional part). Typically, an integer variable may hold values in the range –32,768 to 32,767 and are 2 bytes long.

The ‘char’ specifier

A variable of type char is 1 byte long and is mostly used to hold a single character. For example to declare **grade** to be a character type, you would write:

char grade;

The ‘float’ specifier

It is a type specifier used to declare floating-point variables. These are numbers that have a whole number part and a fractional or decimal part for example 234.936. To declare **Volume to** be of type float, you would write:

float volume;

Floating point variables typically occupy 4 bytes.

The ‘double’ specifier

It is a type specifier used to declare double-precision floating point variables. These are variables that store float point numbers with a precision twice the size of a normal float value. To declare **volume** to be of type double you would write:

double volume;

Double-type variables typically occupy 8 bytes.

* 1. **VARIABLE**

**** A variable is a memory location whose value **can change** during program execution. In C, a variable must be declared before it can be used. Variables can be declared at the start of any block of code.

We have two types of variables: **Local Variable and global Variable**

1. **Local Variable**

Variables declared inside a function are called **local variables.** A local variable is known to and may be accessed by only the function in which it is declared. They are declared and used within a subprogram. The subprogram may be a procedure or a function. Values stored in a local variable can only be used within that function or that procedure.

1. **Global Variable**

Variables declared outside all functions are called **global variables** and they may be accessed by any function in your program. Global variables exist the entire time your program is executing.

They are declared at the top of a program just after the pre-processor directives. They are visible or accessible throughout the whole program.

* + 1. **Variable Names**

 Every variable has a name and a value. The name identifies the variable and the value stores data. There is a limitation on what these names can be. Every variable name in C must start with a letter; the rest of the name can consist of letters, numbers and underscore characters.

C recognizes upper and lower case characters as being different (C is case- sensitive). Finally, you cannot use any of C's keywords like main, while, switch etc as variable names.

**Examples of legal variable names**

x result outfile bestyet

x1 x2 out\_file best\_yet

power impetus gamma hi\_score

It is conventional to avoid the use of capital letters in variable names. These are used for names of constants.

**Illegal Name Rules**

* A variable name cannot start with a digit, asterisk (\*), Arithmetic sign (-,+…), dot (.)and special character (apostrophe)
  + 1. **Declaring Variables**

 **Declaring Variables**

Before a variable can be used it must be declared. Declaring a variable simply means creating that variable. Each variable is associated with one data type. At the point of declaring a variable it must be given a data type.

Syntax for declaring a variable: ***Data type variable\_name***.

**Examples**

int age;

float salary;

double annual\_tax;

double radius;

float circumference;

At the time of declaring a variable it can also be assigned a value immediately e.g int age=30;

Float radius=23. 6;

* 1. **CONSTANTS**

**** A constant is a value that does not change during program execution. In other words, constants are fixed values that may not be altered by the program. A constant should be given an appropriate name and data type associated with it. In C, programming constants are declared in capital letters.

Types of Constants

Constants can be used in C expressions in two ways:

* **Directly**

Here the constant value is inserted in the expression, as it should typically be.

For example:

Area = 3.14 \* Radius \* Radius;

The value **3.14** is used directly to represent the value of **PI** which never requires changes in the computation of the area of a circle.

* **Using a Symbolic Constant**

This involves the use of another C preprocessor, **#define**. For example, **#define PI 3.142**

Example

**A program is required to calculate the area of a circle. Declare the appropriate variables and their correct data types.**

#include <stdio.h>

#define PI 3.14

main()

{

float radius, area;

printf(“Enter the radius of the circle \n”);

scanf(“%f”, &radius);

area = PI \* radius \* radius; /\* PI is a symbolic constant \*/

printf(“Area is %f cm squared “,area);

return 0;

}

* 1. **Operators and Operands**

** Operators and Operands**

An **operator** is a component of any expression that joins individual constants, variables, array elements and function references.

An **operand** is a data item that is acted upon by an operator. Some operators act upon two operands (binary operators) while others act upon only one operand (unary operators).

An operand can be a constant value, a variable name or a symbolic constant.

***Note****: An expression is a combination of operators and operands.*

##### **Examples**

1. x + y ; x, y are operands, + is an addition operator.
2. 3 \* 5; 3, 5 are constant operands, \* is a multiplication operator.
3. x % 2.5; x, 5 are operands, % is a modulus (remainder) operator.
4. sizeof (int); sizeof is an operator (unary), int is an operand.

C include a large number of operators that fall into several kinds of categories.

1. Arithmetic operator
2. Relational Operators
3. Logical Operator
4. Assignment Operator
5. Conditional Operator
   * 1. **Arithmetic Operator**

 **Arithmetic Operator**

There are five arithmetic operators in C.

### **Operator Purpose**

+ Addition

- Subtraction

\* Multiplication

/ Division

% Remainder after integer division

* + 1. **Relational Operator**

 **Relational Operator**

There are four relational operators in C.

* < Less than
* <= Less than or equal to
* > Greater than
* > = Greater than or equal to

Closely associated with the above are two equality operators;

* = = Equal to
* ! = Not equal to
  + 1. **Logical Operator**

 **Logical Operator**

* && Logical AND
* || Logical OR
* ! NOT

The two operators act upon operands that are themselves logical expressions to produce more complex conditions that are either true or false.

* + 1. **Assignment Operator**

 **Assignment Operator**

The most common used assignment operator is ‘=’. It is written as Identifier= Expression

Example: Length=3; Width=5;

* + 1. **Conditional Operator**

 **Conditional Operator**

Conditional tests can be carried out with the conditional operator (**?**). A conditional expression takes the form:

**expression1 ? expression2 : expression3** and implies; evaluate **expression1**. If **expression1** evaluates to **true** ( value is 1 or non zero) then evaluate **expression 2**, otherwise (i.e. if expression 1 is false or zero ) , evaluate **expression3**.

* 1. **Self-test Questions**

****

1. Define the following terms; Variable, constant.
2. Write a program to calculate the volume of:
3. Cuboid
4. Cylinder

To answer Q2 above list all the variables and constants

Declare those variables and constants (symbolic or directly)

* 1. **Summary**

****In this lecture we have discussed about data types, variables, constants and operators and operands. Variables must be declared before they are used in a program. Variable declaration is made by the use of data type associated with that particular variable. In addition, constants are used in programming. The don’t change during program execution. Operators and operands are also a key component in C programming.

* 1. **Reference**

****

1. H.M. Deitel and P.J. Deitel (2014). C: How to Program, 2nd Edition. Prentice Hall, ISBN 341– 7600465.
2. Xavier, C. (2008). *Introduction to Computers and Basic programming*. New Age International Publishers, ISBN 978– 81– 224– 2123– 1.