**Data types**

Data types are used to decide the type and size of variables. C++ is a statically typed programming language. It means, we decide type of variables during the time of compilation.

There are following types of data types available in C++ programming language: -

1. Primitive data types

2. Derived data types

3. User-defined types

**1. Primitive data types: -** Primitive data types are data types which are predefined in C++. It is also known as built-in data types. User can use them directly in the program.

There are following primitive data types available in C++: -

* **Integer: -** Integer data type is used to store numeric value. It takes 2 / 4 bytes in memory to store data. **‘int’** keyword is used to declare a integer data type.
* Range: - (-2147483648 to 2147483467)

**Syntax,**

int variable\_name = value;

**Example,**

int num = 12354;

* **Character: -** Character data type is used to store a single character. It takes 1 byte in memory to store data. **‘char’** keyword is used to declare a character data type.
  + Range: - (-128 to 127)

**Syntax,**

char variable\_name = ‘Value’;

**Example,**

char ch = ‘A’;

* **Float: -** Float data type is used to store decimal value. It can store up to 6 decimal point numbers. It takes 4 bytes in memory to store data. . **‘float’** keyword is used to declare a float data type.
* Range: - (1.4 e -45 to 3.4 e+38)

**Syntax,**

float variable\_name = valuef;

**Example,**

float f1 = 245.250f;

* **Double: -** Double data type is also used to store decimal number. It can store up to 15 decimal point numbers. It takes 8 byte in memory to store data. . **‘double’** keyword is used to declare a double data type.
  + - Range: - (2.3 e -308 to 1.7 e+308)

**Syntax,**

double variable\_name = value;

**Example,**

double d1 = 548998.7894555;

* **Boolean: -** Boolean data type is used to store boolean value which may be either true (1) or false (0). It takes 1 byte in memory to store data. **‘bool’** keyword is used to declare a boolean data type.

**Syntax,**

bool variable\_name = value;

**Example,**

bool b1 = true / false;

* **Void: -** Void means nothing. It does not hold any value. Void data type is used for the function which does not return a value.

**NOTE: -** We can’t declare a variable of void data type.

* **Wide Character: -** Wide character is also used to store a single character. It uses UNICODE table instead of ASCII table. It takes 2 bytes in memory to store data. . **‘wchar\_t’** keyword is used to declare a wide character data type.
  + - Range: - 0 to 65535

**Syntax,**

wchar\_t variable\_name = L‘value’;

**Example,**

wchar\_t chwt = L‘A’;

**NOTE: -** To deal with wide character use wcin or wcout instead of cin or cout.

**Data type modifiers**

Data type modifiers are the keywords which are added to primitive data types as a prefix to modify the basic meaning of primitive data types.

There are following two types of data type modifiers available in C++: -

A. Size data type modifiers

B. Sign data type modifiers

**A. Size data type modifiers: -** Size data type modifiers are used to modify the size of primitive data types.

There are following two types of size data type modifiers available in C++: -

**i. Short: -** Short data type modifier is used to decrease the size of primitive data types. It can be only used with integer data type.

**Syntax,**

short int variable\_name = value;

**‘or’**

short variable\_name = value;

* + - It takes 2 bytes.
    - Range: - (-32768 to 32767)

**ii. Long: -** Long data type modifier is used to increase the size of primitive data types. It can be only used with integer and double data type.

**Syntax,**

long long int variable\_name = value;

‘or’

long long variable\_name = value;

* It takes 8 bytes.
  + - Range: - (-9223372036854775808- to 9223372036854775807)

**Syntax,**

long double variable\_name = value;

* + - It takes 12 bytes.
    - Range: - 1.5e-312 to 1.5e+312

**B. Sign data type modifiers: -** Sign data type modifiers are used to decide that the variable will be store either positive value or negative value. It does not change the size of data types. There are following two types of sign data types modifiers: -

**i. Signed: -** Signed data type modifier is used to store both negative and positive value. It can be only used with char and int data type. By default char and int data types are signed data type.

**Syntax,**

signed char variable\_name = ‘value’;

**‘or’**

char variable\_name = ‘value’;

* Range: - (-128 to 127)

**Syntax,**

signed int variable\_name = value;

**‘or’**

int variable\_name = value;

* Range: - (-2147483648 to 2147483467)

**Syntax,**

signed short int variable\_name = value;

**‘or’**

short int variable\_name = value;

* Range: - (-32768 to 32767)

**Syntax,**

signed long long int variable\_name = value;

**‘or’**

Long long int variable\_name = value;

* + - Range: - (-9223372036854775808- to 9223372036854775807)

**ii. Unsigned: -** Unsigned data type modifier is used to store only positive value. It can be only used with char and int data type. It increases the positive range of data types.

**Syntax,**

unsigned char variable\_name = ‘value’;

* + - Range: - (0 to 255)

**Syntax,**

unsigned int variable\_name = value;

* + - Range: - (0 to 4294967295)

**Syntax,**

unsigned short int variable\_name = value;

* + - Range: - (0 to 65535)

**Syntax,**

unsigned long long int variable\_name = value;

* + - Range: - (0 to 18446744073709551616)

**2. Derived data types: -** Derived data types are data types which are derived from the built-in in data types.

There are following types of derived data types available in C++ programming language: -

* Function
* Array
* Reference
* Pointer

**3. User-defined types: -** User-defined data types are data types which are defined by the user itself.

There are following types of derived data types available in C++ programming language: -

* Class
* Structure
* Union
* Enumeration
* Typedef

**NOTE: -** string is an object of std::string class used to store sequence of characters.

**Syntax,**

string variable\_name = “value”;

**Example,**

string s1 = “Hello C++”;