C - Data Types

Data types in c refer to an extensive system used for declaring variables or functions of different types. The type of a variable determines how much space it occupies in storage and how the bit pattern stored is interpreted.

The types in C can be classified as follows –

The array types and structure types are referred collectively as the aggregate types. The type of a function specifies the type of the function's return value. We will see the basic types in the following section, where as other types will be covered in the upcoming chapters.

Integer Types

The following table provides the details of standard integer types with their storage sizes and value ranges –

To get the exact size of a type or a variable on a particular platform, you can use the **sizeof** operator. The expressions sizeof(type) yields the storage size of the object or type in bytes. Given below is an example to get the size of various type on a machine using different constant defined in limits.h header file –

Live Demo

```
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>
#include <float.h>

int main(int argc, char** argv) {

    printf("CHAR_BIT    :   %d\n", CHAR_BIT);
    printf("CHAR_MAX    :   %d\n", CHAR_MAX);
    printf("CHAR_MIN    :   %d\n", CHAR_MIN);
    printf("INT_MAX     :   %d\n", INT_MAX);
    printf("INT_MIN     :   %d\n", INT_MIN);
    printf("LONG_MAX    :   %ld\n", (long) LONG_MAX);
```

```
    printf("LONG_MIN    :   %ld\n", (long) LONG_MIN);
    printf("SCHAR_MAX   :   %d\n", SCHAR_MAX);
    printf("SCHAR_MIN   :   %d\n", SCHAR_MIN);
    printf("SHRT_MAX    :   %d\n", SHRT_MAX);
    printf("SHRT_MIN    :   %d\n", SHRT_MIN);
    printf("UCHAR_MAX   :   %d\n", UCHAR_MAX);
    printf("UINT_MAX    :   %u\n", (unsigned int) UINT_MAX);
    printf("ULONG_MAX   :   %lu\n", (unsigned long) ULONG_MAX);
    printf("USHRT_MAX   :   %d\n", (unsigned short) USHRT_MAX);

    return 0;
}
```

When you compile and execute the above program, it produces the following result on Linux –

```
CHAR_BIT   :  8
CHAR_MAX   :  127
CHAR_MIN   :  -128
INT_MAX    :  2147483647
INT_MIN    :  -2147483648
LONG_MAX   :  9223372036854775807
LONG_MIN   :  -9223372036854775808
SCHAR_MAX  :  127
SCHAR_MIN  :  -128
SHRT_MAX   :  32767
SHRT_MIN   :  -32768
UCHAR_MAX  :  255
UINT_MAX   :  4294967295
ULONG_MAX  :  18446744073709551615
USHRT_MAX  :  65535
```

Floating-Point Types

The following table provide the details of standard floating-point types with storage sizes and value ranges and their precision –

The header file float.h defines macros that allow you to use these values and other details about the binary representation of real numbers in your programs. The following

example prints the storage space taken by a float type and its range values –

Live Demo

```c
#include <stdio.h>
#include <stdlib.h>
#include <limits.h>
#include <float.h>

int main(int argc, char** argv) {

    printf("Storage size for float : %d \n", sizeof(float));
    printf("FLT_MAX     :   %g\n", (float) FLT_MAX);
    printf("FLT_MIN     :   %g\n", (float) FLT_MIN);
    printf("-FLT_MAX    :   %g\n", (float) -FLT_MAX);
    printf("-FLT_MIN    :   %g\n", (float) -FLT_MIN);
    printf("DBL_MAX     :   %g\n", (double) DBL_MAX);
    printf("DBL_MIN     :   %g\n", (double) DBL_MIN);
    printf("-DBL_MAX     :  %g\n", (double) -DBL_MAX);
    printf("Precision value: %d\n", FLT_DIG );

    return 0;
}
```

When you compile and execute the above program, it produces the following result on Linux –

```
Storage size for float : 4
FLT_MAX     :  3.40282e+38
FLT_MIN     :  1.17549e-38
-FLT_MAX    :  -3.40282e+38
-FLT_MIN    :  -1.17549e-38
DBL_MAX     :  1.79769e+308
DBL_MIN     :  2.22507e-308
-DBL_MAX    : -1.79769e+308
Precision value: 6
```

The void Type

The void type specifies that no value is available. It is used in three kinds of situations –