DATA TYPES IN C LANGUAGE

Data Types

The data type in **C** defines the amount of storage allocated to variables, the values that they can accept, and the operation that can be performed on those variables.

**C** is rich in data types. The variety of data type allow the programmer to select appropriate data type to satisfy the need of application as well as the needs of different machine.

There are three classes of Data-Type

- Primary Data Type
- Derived Data Type
- User Defined Data Type

Primary Data Types (Fundamental Data Types)

All **C** compiler support five type of fundamental data type

1. Integer **int** 2,768 to 32,768
2. Character **char** -128 to 127
3. Floating Point **float** 3.4e-38 to 3.4e+38
4. Double Precision Floating Point **double** 1.7e-308 to 1.7e+308
5. Void Data Type **void** (used for function when no value is to be return)

**Integer Type**

<table>
<thead>
<tr>
<th>Signed</th>
<th>Unsigned</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>unsigned int</td>
</tr>
<tr>
<td>short int</td>
<td>unsigned short int</td>
</tr>
<tr>
<td>long int</td>
<td>unsigned long int</td>
</tr>
</tbody>
</table>

**Character Type**

<table>
<thead>
<tr>
<th>Signed</th>
<th>Unsigned</th>
</tr>
</thead>
<tbody>
<tr>
<td>signed char</td>
<td></td>
</tr>
</tbody>
</table>
### Float Type
- `float`
- `double`
- `long double`

### Void Type
- `void`

*It doesn't return any value*

### Size and Range of Data-Types on a 16-bit machine

<table>
<thead>
<tr>
<th>Types</th>
<th>Size</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>char (or signed char)</td>
<td>8</td>
<td>-128 to 127</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 to 255</td>
</tr>
<tr>
<td>unsigned char</td>
<td>8</td>
<td>0 to 255</td>
</tr>
<tr>
<td>int (or signed int)</td>
<td>16</td>
<td>-32768 to 32767</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 to 65536</td>
</tr>
<tr>
<td>unsigned int</td>
<td>16</td>
<td>0 to 65536</td>
</tr>
<tr>
<td>short int (or signed short int)</td>
<td>8</td>
<td>-128 to 127</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 to 255</td>
</tr>
<tr>
<td>unsigned short int</td>
<td>8</td>
<td>0 to 255</td>
</tr>
<tr>
<td>long int (or signed long int)</td>
<td>32</td>
<td>-2,147,483,648 to 2,147,483,647</td>
</tr>
</tbody>
</table>
unsigned long int
32
0 to 4,294,967,295
float
32
3.4E - 38 to 3.4E + 38
double
64
1.7E - 308 to 1.7E + 308
long
80
3.4E - 4932 to 1.1E + 4932

Note
1. Use sizeof() to know the size of int, char, float etc.

Example program to determine the max and min range of a particular data type.

1. #include<conio.h>
2. #include<stdio.h>
3. void main()
4. {
5. unsigned int i, j;
6. i =1;
7. {
8. j = i;
9. i ++;
10. }
11. printf("Maximum value of unsigned int is = %u", j);
12. printf("Minimum value of unsigned int is = %u", i);
13. }
### Note
1. Do not use increment(++) or decrement(--\) operator with floating points variable.

### A simple program using different data types

1. `#include<conio.h>`
2. `#include<stdio.h>`
3. `void main()`
4. {
5. /*........Declaration part........*/
6. char c;
7. int x, y;
8. float f1, f2;
9. Double d1, d2;
10. unsigned p;
11. /*........Assigning while declaring........*/
12. int a = 4321;
13. long int l = 5432167;
14. /*........Assignment Part .............*/
15. c = 'A';
16. x = 867;
17. f1 = 4.3214;
18. d1 = 8.5467342;
19. f2 = 20.000;
20. d2 = 3.0;
21. /*........Displaying Values.............*/
22. printf("c = %c \n", c);
23. printf("x = %d and y = %d \n", x, y);
24. printf("l = %ld \n", l);
25. printf("d1 = %07lf \n", d1);
26. printf("p = %u \n", p);
27. }

### User define Data type
By using a feature known as "type definition" that allows users to define an identifier that would represent a data type using an existing data type.

**Note**

1. General form:  typedef type identifier;
2. or (to better understand)
3. typedef existing_data_type new_user_define_data_type;

### Example Using user defined data type

1. typedef int number;
2. typedef long big_number;
3. typedef float decimal;
4. typedef double big_decimal;
5. /********* now we can use above user defined types to declare variables*********/
6. number visitors = 25;
7. big_number population = 12500000;
8. decimal radius = 3.5;
9. big_decimal pie = 3.1415926535

**Advantage**

1. The main advantage of typedef is that we can create meaningful data type names for increasing the readability of the program.

### Derived Data Type

Those data types which are derived from fundamental data types are called derived data types. There are basically three derived data types.

1. **Array**: A finite collection of data of same types or homogenous data type.
2. **String**: An array of character type.
3. **Structure**: A collection of related variables of the same or different data types.

**Note**: Details of Array, String and Structure is available separately in this site.

### Examples of derived data types.
1. /*.........Array.........*/
2. int roll_no[40]; //Array to contain roll number of 40 students
3. /*.........String.........*/
4. char name[20] = "Deepak Kumar"; //string of MAX 20 chars
5. /*......... Structure.........*/
6. struct employee
7. {
8.   char name[20];
9.   int age;
10.  float salary;
11. };  // Collection of different data types
12. struct employee emp = {"Shakshi", 28, 450000.00};

Source: http://www.dotnet-tricks.com/Tutorial/c/alat191212-Data-Types-in-C-language.html