What is the size_t data type in C?

size_t is an unsigned integra’ … eader files such as:

```
<stddef.h>, <stdio.h>, <stdlib.h>, <string.h>, <time.h>, <wchar.h>
```

It’s a type which is used to represent the size of objects in bytes and is therefore used as the return type by the **sizeof** operator. It is guaranteed to be big enough to contain the size of the biggest object the host system can handle. Basically the maximum permissible size is dependent on the compiler; if the compiler is 32 bit then it is simply a typedef(i.e., alias) for **unsigned int** but if the compiler is 64 bit then it would be a typedef for **unsigned long long**. The **size_t** data type is never negative.

Therefore many C library functions like *malloc, memcpy and strlen* declare their arguments and return type as **size_t**. For instance,

```
// Declaration of various standard library functions.

// Here argument of 'n' refers to maximum blocks that can be
// allocated which is guaranteed to be non-negative.
void *malloc(size_t n);

// While copying 'n' bytes from 's2' to 's1'
// n must be non-negative integer.
void *memcpy(void *s1, void const *s2, size_t n);

// strlen() uses size_t because the length of any string
// will always be at least 0.
size_t strlen(char const *s);
```

size_t or any unsigned type might be seen used as loop variable as loop variables are typically greater than or equal to 0.

Note: When we use a **size_t** object, we have to make sure that in all the contexts it is used, including arithmetic, we want only non-negative values. For instance, the following program would definitely give the unexpected result:

```
// C program to demonstrate that size_t or
// any unsigned int type should be used
// carefully when used in a loop.
#include<stdio.h>

#define N 10

int main()
{
    int a[N];

    // This is fine.
    for (size_t n = 0; n < N; ++n) {
        a[n] = n;
    }

    // But reverse cycles are tricky for unsigned
    // types as they can lead to infinite loops.
    for (size_t n = N-1; n >= 0; --n)
        printf("%d ", a[n]);
}
```

```
Output
Infinite loop and then segmentation fauilt
```

This article is contributed by Shubham Bansal. If you like GeeksforGeeks and would like to contribute, you can also write an article using contribute.geeksforgeeks.org or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Recommended Posts:

What is data type of FILE in C ?

Is there any need of "long" data type in C and C++?

Bool data type in C++

C++ set for user define data type

Data Type Ranges and their macros in C++

Data type of character constants in C and C++

C++ map having key as a user define data type

Comparison of boolean data type in C++ and Java

Multi-set for user defined data type

Data type of case labels of switch statement in C++?

Conversion of Struct data type to Hex String and vice versa

Difference between Type Casting and Type Conversion

Type Conversion in C

Type Conversion in C++

Type of 'this' pointer in C++

Improved By : jflopezfernandez

Article Tags : C C++ cpp-data-types

Practice Tags : C CPP

👍

31

2.6

□ To-do □ Done

Based on **33** vote(s)

Feedback/ Suggest Improvement Add Notes Improve Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Writing code in comment? Please use ide.geeksforgeeks.org, generate link and share the link here.

Load Comments

A computer science portal for geeks

5th Floor, A-118,
Sector-136, Noida, Uttar Pradesh - 201305
feedback@geeksforgeeks.org