Operator sizeof() in C

In C, sizes of different primitive data types can vary depending on the compiler. If you want to know the size of particular data type in your compiler, C provides the sizeof operator to determine it. It returns the memory required by given variable, data type, or literals in bytes.

Example:

C

```c
#include <stdio.h>
int main()
{
    printf("%zu \n", sizeof(int));
    printf("%zu \n", sizeof(char));
    printf("%zu \n", sizeof(long long));
    printf("%zu", sizeof(float));
    return 0;
}
```

Output

```
4
1
8
4
```

> **Note:** *sizeof() may give different output according to machine, we have run our program on a 32-bit gcc compiler.*

sizeof Operator with Variables and Literals

We can also find size of variables and literals using sizeof operator just like data types.

C

```c
#include <stdio.h>

int main()
{
    int x = 10;
    double d = 11.5;
    printf("%zu \n", sizeof(x));
    printf("%zu \n", sizeof(d));
    printf("%zu \n", sizeof(15LL));
    printf("%zu \n", sizeof(13.6f));
    printf("%zu \n", sizeof(x + 10));
    return 0;
}
```

Output

```
4
8
8
4
4
```

sizeof Operator with Expression

sizeof Operator determines the size of the given variable, literal or expression at compile time and the expressions are not executed. It just deduces the size from the return type of the expression. It is shown in the below example:

C

```c
#include <stdio.h>

int main()
{
    int x = 10;
    printf("%zu \n", sizeof(x++));
    printf("%zu", x);
    return 0;
}
```

Output

```
4
10
```