Format specifier for Integers in C

In C programming, format specifiers are used in functions like **printf** and **scanf** to read and write data. They are used as a placeholder in the format string for input and output with printf, scanf and similar functions. Let's discuss integer format specifiers in C.

There are 8 integer format specifiers in C:

Format Specifier	Data Type	Description
%d	Signed Integer	Used for printing or reading signed integers, supporting both positive and negative values.
%u	Unsigned Integer	Used for storing only positive numbers or zero, effectively doubling the integer range.
%ld	Long Integer	Used for larger integers when normal integers can't hold the required value.
%lld	Long Long Integer	Used for even larger integers, offering a wider range than %ld.
%zd	Size of Data Type	Used for printing size_t values, typically representing the size of objects or arrays.
%x or %X	Hexadecimal Representation	Prints integers in hexadecimal format: %x for lowercase (a-f) and %X for uppercase (A-F).
%o	Octal Representation	Prints integers in octal (base 8) format, with an optional prefix (e.g., 012).
%i	Integer Input	Similar to %d but can read decimal, octal, and hexadecimal numbers. If input starts with 0x or 0, it is treated accordingly.

Example

The below code demonstrates the use of above format specifiers:

C

```c
#include <stdio.h>

int main() {
    int x = 10;
    printf("%d \n", x);

    long y = 20;
```

```
8       printf("%ld \n", y);
9
10      unsigned z = 30;
11      printf("%u \n", z);
12
13      long long w = 40;
14      printf("%lld \n", w);
15
16      size_t u = sizeof(x);
17      printf("%zd \n", u);
18
19      return 0;
20  }
```

Output

```
10
20
30
40
4
```