# GeeksforGeeks

A computer science portal for geeks

GeeksQuiz

Login

| Home | Algorithms | DS | GATE | Interview Corner | Q&A | C | C++ | Java | Books | Contribute | Ask a Q | About |

| Array | Bit Magic | C/C++ | Articles | GFacts | Linked List | MCQ | Misc | Output | String | Tree | Graph |

## Little and Big Endian Mystery

**What are these?**
Little and big endian are two ways of storing multibyte data-types ( int, float, etc). In little endian machines, last byte of binary representation of the multibyte data-type is stored first. On the other hand, in big endian machines, first byte of binary representation of the multibyte data-type is stored first.

Suppose integer is stored as 4 bytes (For those who are using DOS based compilers such as C++ 3.0 , integer is 2 bytes) then a variable x with value 0×01234567 will be stored as following.
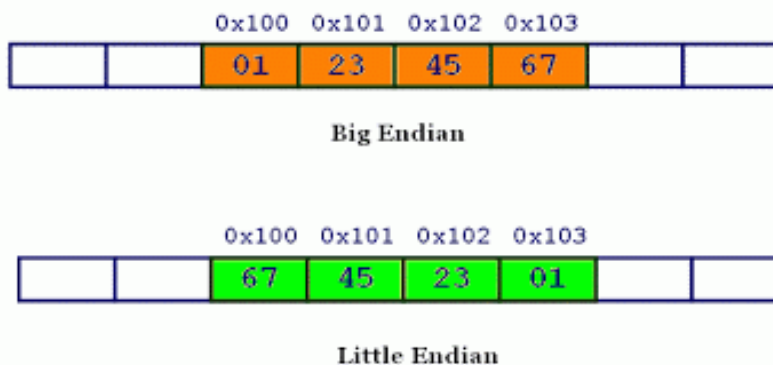


Memory representation of integer ox01234567 inside Big and little endian machines

**How to see memory representation of multibyte data types on your machine?**
Here is a sample C code that shows the byte representation of int, float and pointer.
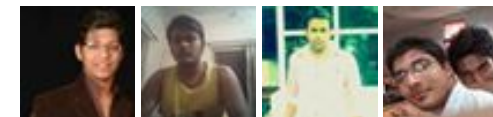
```
#include <stdio.h>
```

---

**GeeksforGeeks**

53,526 people like GeeksforGeeks.

```c
/* function to show bytes in memory, from location start to start+n*/
void show_mem_rep(char *start, int n)
{
    int i;
    for (i = 0; i < n; i++)
         printf(" %.2x", start[i]);
    printf("\n");
}

/*Main function to call above function for 0x01234567*/
int main()
{
    int i = 0x01234567;
    show_mem_rep((char *)&i, sizeof(i));
    getchar();
    return 0;
}
```

When above program is run on little endian machine, gives "67 45 23 01″ as output , while if it is run on endian machine, gives "01 23 45 67″ as output.

**Is there a quick way to determine endianness of your machine?**
There are n no. of ways for determining endianness of your machine. Here is one quick way of doing the same.

```c
#include <stdio.h>
int main()
{
    unsigned int i = 1;
    char *c = (char*)&i;
    if (*c)
         printf("Little endian");
    else
         printf("Big endian");
    getchar();
    return 0;
}
```

In the above program, a character pointer c is pointing to an integer i. Since size of character is 1 byte when the character pointer is de-referenced it will contain only first byte of integer. If machine is little endian then *c will be 1 (because last byte is stored first) and if machine is big endian then *c will be 0.

## Popular Posts

**Does endianness matter for programmers?**

Most of the times compiler takes care of endianness, however, endianness becomes an issue in following cases.

It matters in network programming: Suppose you write integers to file on a little endian machine and you transfer this file to a big endian machine. Unless there is little andian to big endian transformation, big endian machine will read the file in reverse order. You can find such a practical example here.

Standard byte order for networks is big endian, also known as network byte order. Before transferring data on network, data is first converted to network byte order (big endian).

Sometimes it matters when you are using type casting, below program is an example.

```c
#include <stdio.h>
int main()
{
    unsigned char arr[2] = {0x01, 0x00};
    unsigned short int x = *(unsigned short int *) arr;
    printf("%d", x);
    getchar();
    return 0;
}
```

In the above program, a char array is typecasted to an unsigned short integer type. When I run above program on little endian machine, I get 1 as output, while if I run it on a big endian machine I get 256. To make programs endianness independent, above programming style should be avoided.

**What are bi-endians?**

Bi-endian processors can run in both modes little and big endian.

**What are the examples of little, big endian and bi-endian machines ?**

Intel based processors are little endians. ARM processors were little endians. Current generation ARM processors are bi-endian.

Motorola 68K processors are big endians. PowerPC (by Motorola) and SPARK (by Sun) processors were big endian. Current version of these processors are bi-endians.

**Does endianness effects file formats?**

File formats which have 1 byte as a basic unit are independent of endianness e..g., ASCII files . Other file formats use some fixed endianness forrmat e.g, JPEG files are stored in big endian format.

**Which one is better — little endian or big endian**

The term little and big endian came from Gulliver's Travels by Jonathan Swift. Two groups could not agree by which end a egg should be opened -a-the little or the big. Just like the egg issue, there is no technological reason to choose one byte ordering convention over the other, hence the arguments degenerate into bickering about sociopolitical issues. As long as one of the conventions is selected and adhered to consistently, the choice is arbitrary.

## Related Tpoics:

- Analysis of Algorithms | Set 4 (Analysis of Loops)
- Analysis of Algorithms | Set 3 (Asymptotic Notations)
- NP-Completeness | Set 1 (Introduction)
- Static and Dynamic Libraries | Set 1

## Recent Comments

Abhi You live US or India?

Google (Mountain View) interview · 20 minutes ago

**Aman** Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 59 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 1 hour ago

**Sanjay Agarwal** bool tree::Root_to_leaf_path_given_sum(tree...

Root to leaf path sum equal to a given number · 1 hour ago

**GOPI GOPINATH** @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number · 1 hour ago

**newCoder3006** If the array contains negative numbers also. We...

Find subarray with given sum · 1 hour ago

► Byte

► Int

- [The Ubiquitous Binary Search | Set 1](#)
- [Reservoir Sampling](#)
- [Analysis of Algorithms | Set 2 (Worst, Average and Best Cases)](#)
- [Analysis of Algorithms | Set 1 (Asymptotic Analysis)](#)

 22    **Tweet** 1          2

**Writing code in comment?** Please use **ideone.com** and share the link here.

## 14 Comments          **GeeksforGeeks**

Sort by Newest ▼

Join the discussion…

**bonnyswan** · a month ago
Thanks for sharing. I was looking to find some clear explanation with examples
∧ | ∨ ·

**akshay** · 9 months ago
Thank you. Nicely explained the concept of Endianness.
1 ∧ | ∨ ·

**Ravi Kumar** · a year ago
thank u for the info guys................
∧ | ∨ ·

**BackBencher** · a year ago
@All:
if i take num = 256
it will print Big Endian, Please explain me why this, i am bit confuse to endines
∧ | ∨ ·

**GeeksforGeeks** · a year ago

@abhishek08aug and @amitp49: Thanks for pointing this out. We have fixed t

∧ | ∨ ·

**abhishek08aug** · a year ago

Are not the definitions here for both big and little endian exactly same?

In little endian machines, last byte of binary representation of the multibyte dat
hand, in big endian machines, first byte of binary representation of the multibyt

Please fix it as below:

In little endian machines, last byte of binary representation of the multibyte dat
hand, in big endian machines, first byte of binary representation of the multibyt

```
/* Paste your code here (You may delete these lines if not writing c
```

∧ | ∨ ·

**amitp49** · 2 years ago

I think there is typo error in definition...

"Little and big endian are two ways of storing multibyte data-types ( int, float, e
of binary representation of the multibyte data-type is stored first. On the other
of binary representation of the multibyte data-type is stored last."

It should be..

"Little and big endian are two ways of storing multibyte data-types ( int, float, e
of binary representation of the multibyte data-type is stored first. On the other
of binary representation of the multibyte data-type is stored last."

Correct me if i am wrong...

**KK123** · 3 years ago

Thats not called as typo ha ha... anyway thats bearable as compared to quality

ᵔ | ᵛ ·

**Algoseekar** · 3 years ago

What Little-Endian and Big-Endian? How can I determine whether a machine's
endian? How can we convert from one to a
First of all, Do you know what Little-Endian and Big-Endian mean?

Little Endian means that the lower order byte of the number is stored in memo
higher order byte is stored at the highest address. That is, the little end comes

For example, a 4 byte, 32-bit integer

Byte3 Byte2 Byte1 Byte0

will be arranged in memory as follows:

Base_Address+0 Byte0
Base_Address+1 Byte1
Base_Address+2 Byte2
Base_Address+3 Byte3

Intel processors use "Little Endian" byte order.

see more

ᵔ | ᵛ ·

**Patil** ➜ Algoseekar · a year ago

@Algoseeker :
if i take num = 256
it will print Big Endian, Please explain me why this, i am bit confuse to

```
        /* Paste your code here (You may **delete** these lines **if not** wri
```

**Venki** · 4 years ago

All the examples cited above tell endian-ness during runtime. Is there any way

**Felipe Pena** → Venki · 3 years ago

A bit late, but see http://unixpapa.com/incnote/by... :)

**sankalp** · 4 years ago

```c
#include<stdio.h>
int main()
{
    unsigned char arr[2] = {0x01, 0x00};
    unsigned short int x = *(unsigned short int *) arr;
    printf("%d", x);
    getchar();
    return 0;
}
```

In this question output in big endian system should be 256 instead of 512

**GeeksforGeeks** → sankalp · 4 years ago

@sankalp: Thanks for pointing this out, we have corrected the typo.

@geeksforgeeks, **Some rights reserved**          **Contact Us!**                    Powered by **WordPress** & **MooTools**, customized by geeksforgeeks team