Position of rightmost set bit

Write a one line C function to return position of first 1 from right to left, in binary representation of an Integer.

```
I/P    18,   Binary Representation 010010
O/P   2
I/P    19,   Binary Representation 010011
O/P   1
```

Let I/P be 12 (1100)

```
Algorithm: (Example 18(010010))

1. Take two's complement of the given no as all bits are reverted
except the first '1' from right to left (10111)

2  Do an bit-wise & with original no, this will return no with the
required one only (00010)

3  Take the log2 of the no, you will get position -1 (1)

4  Add 1 (2)
```

Program:

```
#include<stdio.h>
#include<math.h>

unsigned int getFirstSetBitPos(int n)
```


```
{
    return log2(n&-n)+1;
}

int main()
{
     int n = 12;
     printf("%u", getFirstSetBitPos(n));
     getchar();
     return 0;
}
```


Related Tpoics:

- Check if a number is multiple of 9 using bitwise operators
- How to swap two numbers without using a temporary variable?
- Divide and Conquer | Set 4 (Karatsuba algorithm for fast multiplication)
- Find position of the only set bit
- Swap all odd and even bits
- Add two bit strings
- Write your own strcmp that ignores cases
- Binary representation of a given number

Popular Posts

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

38 Comments GeeksforGeeks

Sort by Newest

Join the discussion…

Abhishek • 3 months ago
What if n=0;
• Reply • Share ›

RAUL • 4 months ago
One line code:

while (!(n & 1) && ++pos && (n >>= 1));

cout << pos+1;
• Reply • Share ›

divyank_duvedi • 4 months ago
Another Method to do it :
XOR=num^(num-1);
pos_rightset=(log(XOR)/log(2))+1;
• Reply • Share ›

Ujjwal Arora • 10 months ago

```
log2((n & (n-1)) ^ n)) + 1;
```

Reply • Share ›

olive • 10 months ago

```
int getrmostbit(int n)
{count=0;
while(n!=0)
{
if(n&1)
{count++;break;}
count++;
n=n>>1;

}
return count;
}
```

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

Reply • Share ›

crazy • 11 months ago

why not this..

```
return n&~(n-1);
```

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

Reply • Share ›

Recent Comments

Abhi You live US or India?

Google (Mountain View) interview · 27 minutes ago

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

Write a C program to Delete a Tree. · 1 hour 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 · 2 hours ago

I am sorry it will not work....

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

• Reply • Share ›

Sandeep Yadav • 11 months ago

another solution in O(n) time where n is number of bits in n.
1.initiate b=1, c=1; //n is number for which set bit is be find.
2.while(!(b&n))
a.shift left b.
b.increment c.
3.print c;
programe for implementation is.

```
#include<iostream>
using namespace std;.
int main()
{
int n, b=1, c=1;
cin>>n;
while(!(n&b))
{
b<<=1;
c++;
}
cout<<c<<endl;
return 0;
}
```

• Reply • Share ›

minoz • 11 months ago

Seems a typo in the example.

naive • a year ago

#include <stdio.h>

int main()
{
int n,i,set=0;
printf("Enter the number:-");
scanf("%d",&n)
int mask = 1;
for(i=1;i<=32;i++)
{
if((mask & n) == 1) { set = 1 break; }
else n = n >> 1;
}

if(set == 1) printf("%d",i);
else printf("No Set Bit Found !!!");
return 0;
}

Ranjan • a year ago

prints the binary value of the given decimal number along with the first set bit p

```
 /* Paste your code here (You may delete these lines if not writing c
int bin(unsigned n)
{
    unsigned i;
    int count =-1;
```

```
    for (i = 1<<31 ; i > 0; i = i / 2)
    {
         (n & i)? printf("1"): printf("0");
          if(n & i)
         {
               count = i/2;
         }
    }
    return count+1;
}
```

see more

• Reply • Share ›

Amandeep Sharma • a year ago

Method you have shown above uses log function, which implicitly uses multipi
method.

```
 #include<stdio.h>
#include<conio.h>
int main()
{
    int num=16;
    int mask=1;
    int pos;
    if(num==0)
    {
        printf("0");
        return 0;
    }
    for(pos=1;!(mask & num);pos++)
```

```
    {
        mask=mask<<1;
    }
    printf("position : %d ",pos);
    return 0;
}
```

Reply • Share ›

Monidipa Chakraborty • a year ago

```
 int rightmost_setbit(int n)
{
     return (n&(!(n-1)));
}
```

Reply • Share ›

pr6989 • 2 years ago

```
 #include<iostream>
#include<math.h>
using namespace std;
main()
{
    int n;
    cin>>n;
    if(n==0)
    cout<<0;
    else
    {
        int check=1;
        while(check<n)
```

```
        {
            if(check&n)
            break;
            else
            check<<=1;
        }
        cout<<log2(check)+1;
    }

}
```

Reply • Share ›

pr6989 ➦ pr6989 • 2 years ago

The above code works for positive numbers only.

Reply • Share ›

shivi • 2 years ago

can't just
n-(n&(n-1))
give the answerr?

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

Reply • Share ›

vick • 2 years ago

my solution

```
/*
#include<stdio.h>
```

```
#include<conio.h>

int posOf1st1(int a,int *c)
{
        for((*c)=1;!(a & 1);(*c)++,a=a>>1);
};

int main()
{
        int a,c;
        scanf("%d",&a);
        posOf1st1(a,&c);
        printf("%d",c);
        getch();
} */
```

Reply • Share ›

bhavneet • 2 years ago

why not this?

```
 int func(int n)
{
return log2(n-n&(n-1));
}
```

Reply • Share ›

akshat gupta bhavneet • 11 months ago

it is perfectly fine.

Reply • Share ›

Neha • 2 years ago

Apply n & !(n-1)

^ | v • Reply • Share ›

Rudhi • 2 years ago

/* comment on my simple logic */
include

int position_of_rightmost_set_bit (int);

int
main ()
{
int enter_num;
printf ("enter an integer \n");
scanf ("%d", &enter_num);
printf ("The position of the rightmost set bit is equal to %d",
position_of_rightmost_set_bit(enter_num));
return 0;
}

int
position_of_rightmost_set_bit (int num)
{

see more

^ | v • Reply • Share ›

Rudhi • 2 years ago

/* Hi!! Rudhi here Plz comment me on this piece of code */

```
# include

int position_of_rightmost_set_bit (int);

int
main ()
{
int enter_num;
printf ("enter an integer \n");
scanf ("%d", &enter_num);
printf ("The position of the rightmost set bit is equal to %d",
position_of_rightmost_set_bit(enter_num));
return 0;
}
```

see more

Reply • Share ›

levis • 2 years ago

log2(n^(n&n-1))+1

Reply • Share ›

Piyush levis • 11 months ago

elaborate

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

Reply • Share ›

Abhimanyu Vohra • 2 years ago

nice!!

Reply • Share ›

Kk • 3 years ago

```
while(!(n & (1 << (i++))));
//i points to position, but it is O(b), where b is the no. of bits in
```

• Reply • Share ›

rajat • 3 years ago

```
int first_set_bit(int n){
        /* I have assumed LSB is at position 0 and 2nd bit is at posi
      int first_bit= (n&(~(n-1)))/2;
      return(first_bit +1);
}
```

• Reply • Share ›

BlackMath ➦ rajat • 2 years ago

Fails for input 8, 16, etc.

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

• Reply • Share ›

Himanshu • 3 years ago

Another implementation used by Linux kernel is :

A function ffs() is defined as:

```
/**
  * ffs - find first bit set
   * @x: the word to search
   *
   * This is defined the same way as
```

```
    * the libc and compiler builtin ffs routines, therefore
    * differs in spirit from the above ffz (man ffs).
    */
  static inline int ffs(int x)
  {
          int r = 1;

          if (!x)
                  return 0;
          if (!(x & 0xffff)) {
```

see more

• Reply • Share ›

Hill • 3 years ago

```
 #include<iostream>
#include<stdio.h>

using namespace std;
int main()
{
int pos=1;

unsigned int v=18;
unsigned int m=1;

while (v)
{
    if(v&m) break;
    pos++;
    v >>=1;
```

```
}
cout<<pos;
getchar();
}
```

Reply • Share ›

guineaPig • 3 years ago

```
#include<stdio.h>
int place(int n) {
        int pos=1,num=1;
        while(!(n & num)) {
                num<<=1;
                pos++;
        }
       return pos;
}
int main()
{
        int n=64;
        printf("%d",place(64));
        getchar();


        return 0;
}
```

Reply • Share ›

Venki • 4 years ago

An order of log(X) algorithm. We can conclude from 2's complement form that complement form by complementing each bit from right most set bit". For exa

following way (assuming 8 bit size)

7 = 00000111
-7 = 11111001

8 = 00001000
-8 = 11111000

5 = 00000101
-5 = 11111011

If we perform ANDing between x and -x we left with right most set bit. All this ta
[O(log(x))] to figure out which bit is set. Given below is code.

```
int getPosition(unsigned x)
{
    // ASSUMPTION: x will not be zero
```

see more

• Reply • Share ›

wgpshashank Venki • 2 years ago
nice venki , thumbs up !!!

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

• Reply • Share ›

Abhishek • 4 years ago
Sorry...it shud be right shift not left shift in the previous post

• Reply • Share ›

Abhishek • 4 years ago

I wud like to suggest another solution...plz tell me if i m wrong

Let the given number be n..

m= n^(n-1)
now left shift m and keep increasing the counter by 1.
when m becomes 0, the value of count is the required value..

amit • 4 years ago

is there any efficient method for position of leftmost bit????

Hary • 4 years ago

Game your solution is awesome. But this will not work for n = 0 and it also ass
returned will be 0.

game • 4 years ago

Using 'log' is never a good idea, iterating over the bits will be much more effici
highly costly operations because they generally include memory accesses.
Below is a ~40-50 instruction O(1) solution

```
 #include<stdio.h>
unsigned int getFirstSetBitPos(int n)
{
  int b =0;
  n--;
  if((n&0xffff) == 0xffff)
  {
    b+=16;
    n>>=16;
```

```
if((n&0xff) == 0xff)
{
  b+=8;
  n>>=8;
```

see more

• Reply • Share ›

Subscribe Add Disqus to your site