

Forum:[Hack Forums](#)

Thread: [Coding Contest - Rotating Bits](#)

g4143 - June 2, 2009, 3:46 pm

Create a program/function that will rotate the bits of a unsigned integer by n times in the left or right direction...

So what is rotating bits, well lets look at an example:?

Say we have a binary number 0110 0101 and we want to rotate this number to the right by two. First thing we do is shift the number to the right by two pushing two numbers off 0110 01 - 01 then we take those two numbers 01 and place them on the left of the binary number so the result is 0101 1001.

So you can see anything we push off the right side is picked up and placed on the left. Its kind of like a

conveyor belt with bits. Now to rotate to the left we just reverse the process...

So to return to the task - create a program/function that will rotate a unsigned integer to the left or right n times...

Note it will help if you have a calculator that displays numbers and binary...G4143

This is the last one...Its officially someone else's turn to come with the next task or competition

JapaBrz - June 2, 2009, 5:48 pm

in vb.net

Code Module Module1

```
Public Function GetBinary(ByVal S As UInteger) As String ' letter by letter
```

```
    Dim r As UInteger
```

```
    Dim i As Integer
```

```
    r = S
```

```
    For i = 0 To S
```

```
        If r < 2 ^ i Then
```

```
            Exit For
```

```
        End If
```

```
        r -= 2 ^ i
```

```
    Next
```

```
    Dim k As Long
```

```
    k = 2 ^ (i + 1)
```

'this part gets the bit size of the number. its to make sure that you wont put useless loops

```
    While k <> 0
```

```
        If S >= k Then
```

```
            S = S - k
```

```
            GetBinary = GetBinary + "1"
```

```
        Else
```

```
            GetBinary = GetBinary + "0"
```

```
        End If
```

```
        k = k - 2 ^ i
```

```
        i = i - 1
```

```

End While
'reads from top to down, if the number fits on the bit size, remove the number from the total and add
a 1, else add a 0
For f As Integer = Len(GetBinary) To i + 1
    GetBinary += "0"
Next
End Function
'if there is many 000 missed on the right side(thats why i made the first loop) only fill them.
Public Function getUInteger(ByVal Binary As String) As UInteger
    For i As Integer = 1 To Binary.Length
        If Mid(Binary, i, 1) = "1" Then
            getUInteger += 2 ^ (Binary.Length - i)
        End If
    Next
End Function
'converts back to uinteger
Public Function Rotate(ByVal Binary As String, ByVal N As Integer, ByVal dLeft As Boolean,
ByVal amountOfN As Integer) As String
    Dim i As Integer
    Dim buffer As String = Binary
    If dLeft Then
        While i < N
            Binary = Right(Binary, amountOfN) + Left(Binary, Binary.Length - amountOfN)
            i += 1
        End While
    Else
        While i < N
            Binary = Right(Binary, Binary.Length - amountOfN) + Left(Binary, amountOfN)
            i += 1
        End While
    End If
    Return Binary
End Function
Sub Main()
    Console.WriteLine(getUInteger(Rotate(GetBinary(121), 1, False, 2)))
    Console.ReadKey()
End Sub

```

End Module
here , takes a uinteger, converts to binary, and rotates it to left or right.

Calls

getbinary(value as uinteger) as string

converts the uinteger value to a binary string

Rotate(ByVal Binary As String, ByVal N As Integer, ByVal dLeft As Boolean, nAmount as integer) As String

takes a binary string, rotate N times to left(is true) or right(if false) and returns the binary string

wanna convert back to uinteger, namount is the reversed chunk size

[edited]

this function converts back to uinteger

Code Public Function getUInteger(ByVal Binary As String) As UInteger

```

For i As Integer = 1 To Binary.Length
    If Mid(Binary, i, 1) = "1" Then
        getUInteger += 2 ^ (Binary.Length - i)
    End If
Next
End Function

```

binary is the binary string and returns the uinteger value
the runcode ill be

```

getUInteger(Rotate(GetBinary(121), 1, False, 2))
01111001
11100101

```

returns this

g4143 - June 2, 2009, 6:09 pm

japabrz,

I don't have VB so I have no idea if you program is right or wrong so I'll provide you with a number:

3222269959

if you rotate this to the right 4 bits you should arrive at a number:

2080440064

This is assuming my solution is correct...G4143

g4143 - June 3, 2009, 4:40 am

What only one taker?...G4143

g4143 - June 3, 2009, 8:57 am

The solution - or my solution...G4143

```

Code #include <stdio.h>
#include <stdlib.h>

```

```

#define LEFTRIGHT int
#define HIGHMASK 0x80000000
#define LOWMASK 0x00000001

```

```

enum {LEFT, RIGHT};

```

```

int rotateit(unsigned int x, int ntime, LEFTRIGHT LR)
{
    int i = 0;
    if (LR == LEFT)
    {
        for (i = 0; i < ntime; ++i)
        {
            if (x & HIGHMASK)
            {
                x = (x<<1) + LOWMASK;
            }
        }
    }
}

```

```

    }
    else
    {
        x = (x<<1);
    }
}
}
else
{
    for (i = 0; i < ntime; ++i)
    {
        if (x & LOWMASK)
        {
            x = (x>>1) + HIGHMASK;
        }
        else
        {
            x = (x>>1);
        }
    }
}
return x;
}

int main(int argc, char**argv)
{
    unsigned int ans = rotateit(3222269959, 4, RIGHT);
    fprintf(stdout, "ans->%u\n", ans);
    ans = rotateit(3222269959, 4, LEFT);
    fprintf(stdout, "ans->%u\n", ans);
    exit(EXIT_SUCCESS);
}

```

JapaBrz - June 3, 2009, 5:10 pm

my version only works with unassigned 16bit integer values. what it does:

- convert the integer to binary
- gets the last N amount of bits from right and puts to left or
gets the last N amount of bits from left and puts to right Ni times
- converts back to uinteger

g4143 - June 3, 2009, 5:23 pm

(Jun 3, 2009 05:10 PM)japabrz Wrote: [\[CITE\]](#) -convert the integer to binary

Now I know this is nick picking but the number is already in binary. The software just displays it in a format we are more comfortable with, say decimal, so you really don't have to convert it just apply the proper operators/functions to it since its conveniently already in binary...G4143

JapaBrz - June 3, 2009, 5:35 pm

omg i know that 00000001=1 and 10000000=128 -_-

32bit 16 8 , etc. im saying what the function i wrote uses a different method cuz i dont find any other way of moving the last bits of a uint value in vbnet without converting to binary first. i know is possible to with some other operation but since i already have a binary function(that i wrote) i use it to save time

so lets say that i mean is
"Translate" the integer to Binary ^^

g4143 - June 3, 2009, 5:47 pm

(Jun 3, 2009 05:35 PM)japabrz Wrote: [\[CITE\]](#) omg i know that 00000001=1 and 10000000=128 -_- 32bit 16 8 , etc. im saying what the function i wrote uses a different method cuz i dont find any other way of moving the last bits of a uint value in vbnet without converting to binary first. i know is possible to with some other operation but since i already have a binary function(that i wrote) i use it to save time

so lets say that i mean is
"Translate" the integer to Binary ^^

This is one of the reasons I love C. Everything is out in the open, the walls of abstraction are very thin here, actually see through in most cases...G4143

Note I'm not saying that C is a better language because it can be horrendous at times. Its just where I have to be in programming - in control, maybe after a few more years with low/mid-level languages I'll be comfortable enough to make the step to a higher level language..G4143

scaryblackguy - June 3, 2009, 9:21 pm

(Jun 3, 2009 05:47 PM)g4143 Wrote: [\[CITE\]](#) maybe after a few more years with low/mid-level languages I'll be comfortable enough to make the step to a higher level language..G4143

lol. This is the most backwards statement ever. However, I think more people should think like you. Learn how the computer works with low level programming such as C and ASM, and then move up to the higher level languages once you know how the functions are actually working. This probably makes your code more efficient and powerful. I wish I took the same programming route as you.

g4143 - June 3, 2009, 9:54 pm

(Jun 3, 2009 09:21 PM)scaryblackguy Wrote: [\[CITE\]](#) lol. This is the most backwards statement ever. However, I think more people should think like you. Learn how the computer works with low level programming such as C and ASM, and then move up to the higher level languages once you know how the functions are actually working. This probably makes your code more efficient and powerful. I wish I took the same programming route as you.

One thing a lot of people do is assume that since you can program you understand how computers works. Programming is just a step, learning operating systems is another and hardware another, networking - internet - graphics - etc after enough steps your a competent hacker....G4143

scaryblackguy - June 3, 2009, 10:03 pm

(Jun 3, 2009 09:54 PM)g4143 Wrote: [\[CITE\]](#) One thing a lot of people do is assume that since you can program you understand how computers works. Programming is just a step, learning operating systems is another and hardware another, networking - internet - graphics - etc after enough steps your a competent hacker....G4143

very true, very true. Programming is indeed a very large step, but overall the bigger picture is the most important. You're a pretty wise guy g4143