PART-1

BIT MANIPULATION OPERATORS AND LEVEL-1 PROGRAMS

Shivaleela Hoonalli

Level-1 Programs

1. General program to demonstrate bitwise operators.
2. Check if the ‘n’ th bit is set or not.
3. Write a program to set nth bit of a number.
4. Write a program to clear nth bit of a number.
5. Write a program to toggle nth bit of a number
6. Write a program to toggle ‘n’ bits from given position of a number.
7. Write a program to set ‘n’ bits from given position of a number.
8. Write a program to get ‘n’ bits a of number from LSB
9. Write a program to get ‘n’ bits from given position of a number.
10. Write a program to replace ‘n’ bits of given number.

Operators

1.**Bitwise &:**It takes two operands as inputs, does the and operation on each bit of two numbers. The result of & operation is '1' only if both the bits are 1,otherwise '0'.

2.**Bitwise |:**It takes two operands as input,does the or operation on every bit of two operands.The result of | operation is '0' if both the bits are '0',otherwise '1'.

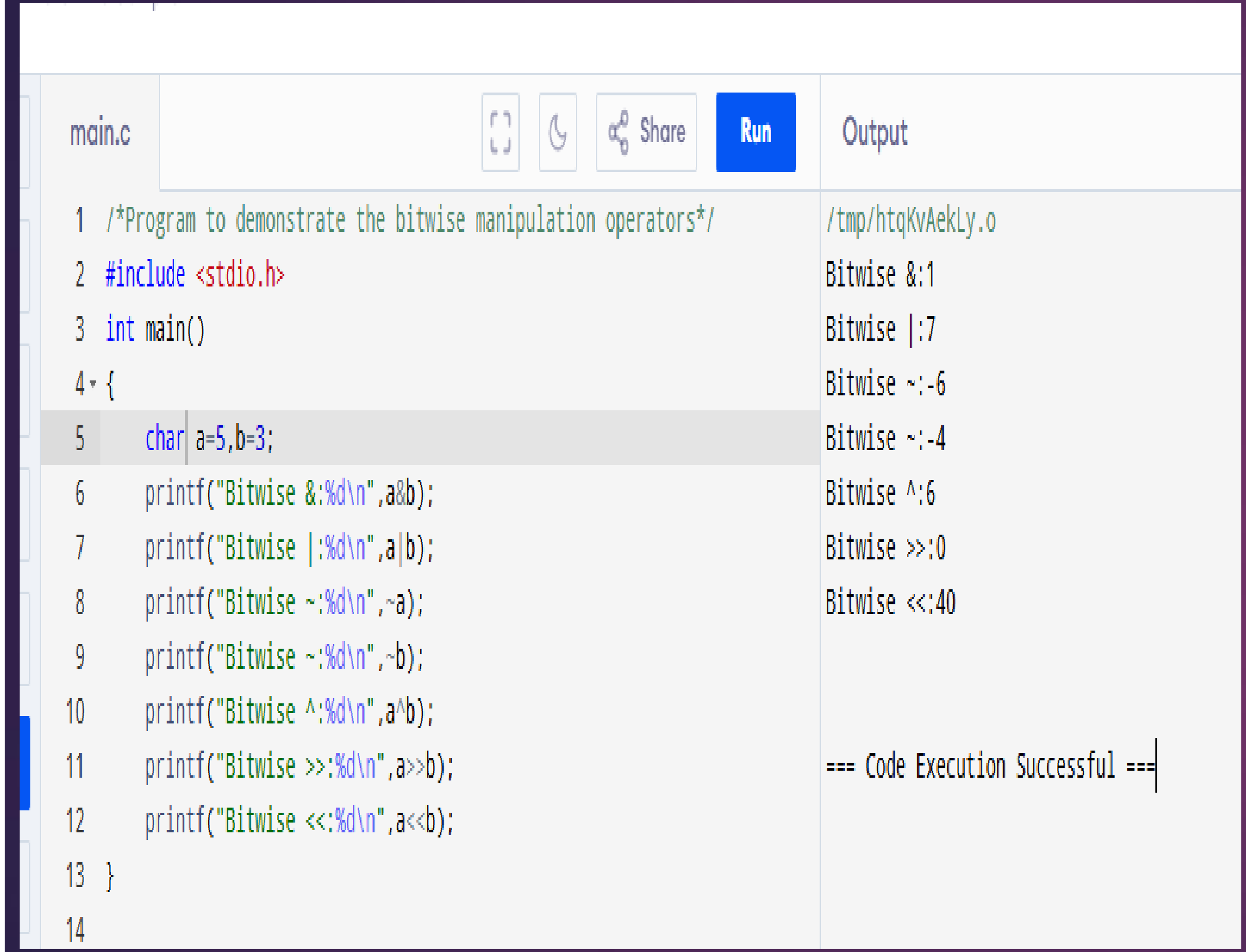
3.**Bitwise ~:**It takes one operand as a input,does the invert operation on each bit of the number.

4.**Bitwise ^:**It takes two operands as input,does the xor operation on every bit of two numbers.The result of the operation is '1' if both the bits are different, otherwise '0'.

5.**Bitwise right shift(>>):**It takes the two operands as a input, right shift the bits of first operand,second operand decides by how many bit positions we need to right shift the first operand.

6.**Bitwise left shift(<<):**It takes the two operands as a input, left shift the bits of first operand,second operand decides by how many bit positions we need to left shift the first operand.

1. General program to demonstrate bitwise operators.



```c
/*Program to demonstrate the bitwise manipulation operators*/
#include <stdio.h>
int main()
{
    char a=5,b=3;
    printf("Bitwise &:%d\n",a&b);
    printf("Bitwise |:%d\n",a|b);
    printf("Bitwise ~:%d\n",~a);
    printf("Bitwise ~:%d\n",~b);
    printf("Bitwise ^:%d\n",a^b);
    printf("Bitwise >>:%d\n",a>>b);
    printf("Bitwise <<:%d\n",a<<b);
}
```

```
/tmp/htqKvAekLy.o
Bitwise &:1
Bitwise |:7
Bitwise ~:-6
Bitwise ~:-4
Bitwise ^:6
Bitwise >>:0
Bitwise <<:40


=== Code Execution Successful ===
```

Explanation

a=5,b=5

a=00000101

b=00000011

a&b= 00000001=1

a|b= 00000111=7

a^b= 00000110=6

~a=11111010-a is signed char…msb is 1 so number is negative..take 2's complement of a number→00000101+1=00000110→ -6

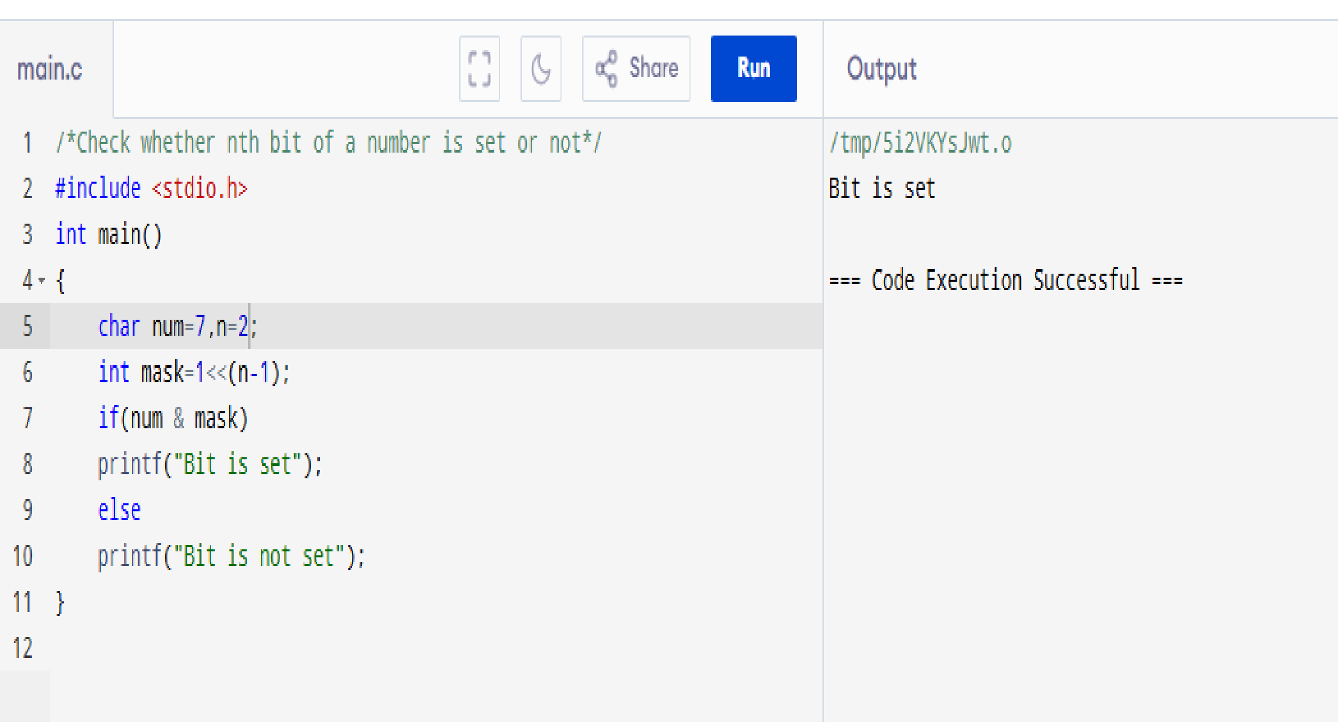
~b=11111100→b is signed char..msb is 1 so number is negative..take 2's complement of a number→00000011+1=00000100→ -4

a>>b=00000101>>3→00000000→0

a<<b=00000101<<3→00101000→40

2. Check if the ‘n’ th bit of a number is set or not.

Let’s have number as 7 and check 2nd bit is set or not



```c
/*Check whether nth bit of a number is set or not*/
#include <stdio.h>
int main()
{
    char num=7,n=2;
    int mask=1<<(n-1);
    if(num & mask)
    printf("Bit is set");
    else
    printf("Bit is not set");
}
```

```
/tmp/5i2VKYsJwt.o
Bit is set


=== Code Execution Successful ===
```

Explanation:

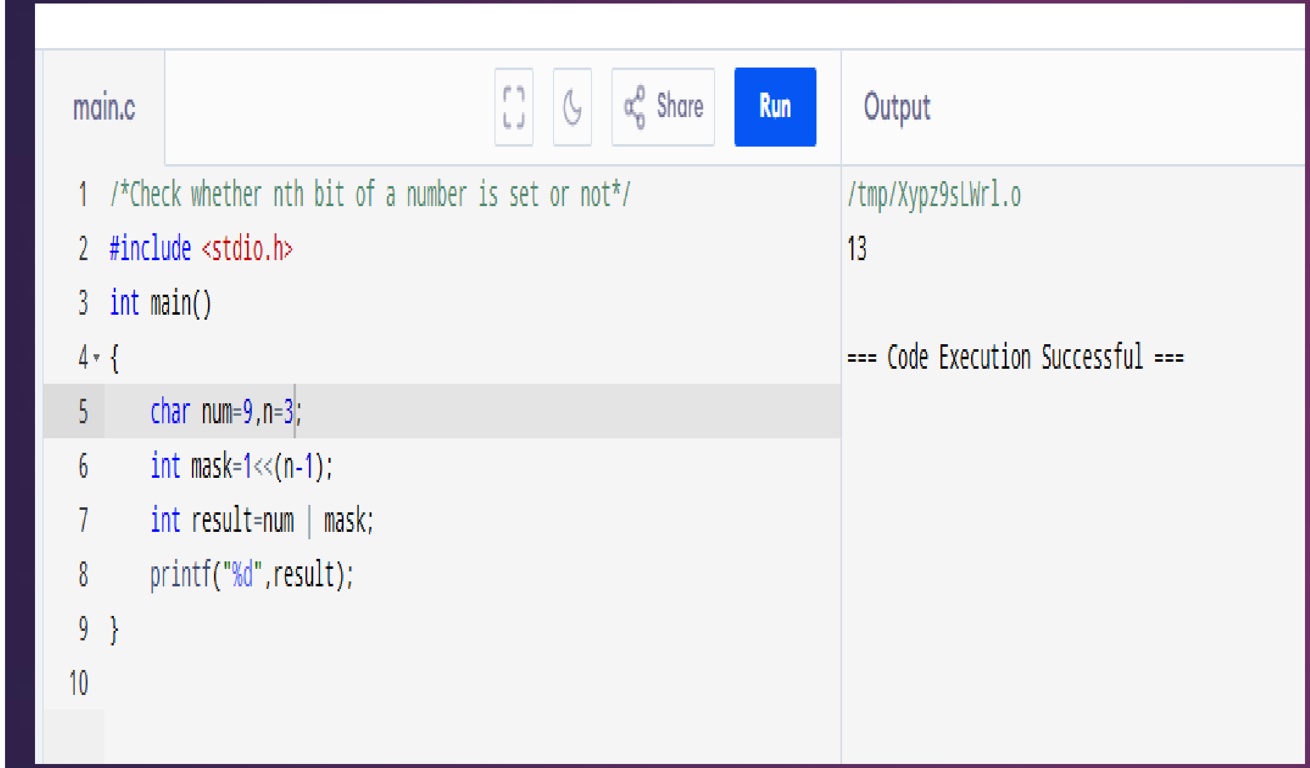
Num=7→0 0 0 0 0 1 1 1

3rd 2nd 1st

Num: 00001001
Mask :00000010
Mask=1<<(n-1)→1<<(2-1)→1<<1→00000010
result=num&mask→00000111 & 00000010→2(nonzero)→so bit is set.

3. Write a program to set nth bit of a number.

Let's have number as 9 and set 3rd bit.



```c
/*Check whether nth bit of a number is set or not*/
#include <stdio.h>
int main()
{
    char num=9,n=3;
    int mask=1<<(n-1);
    int result=num | mask;
    printf("%d",result);
}

```

```
/tmp/Xypz9sLWrl.o
13


=== Code Execution Successful ===
```

Explanation:

Num=9→0 0 0 0 1 0 0 1

3rd 2nd 1st

Num: 00001001

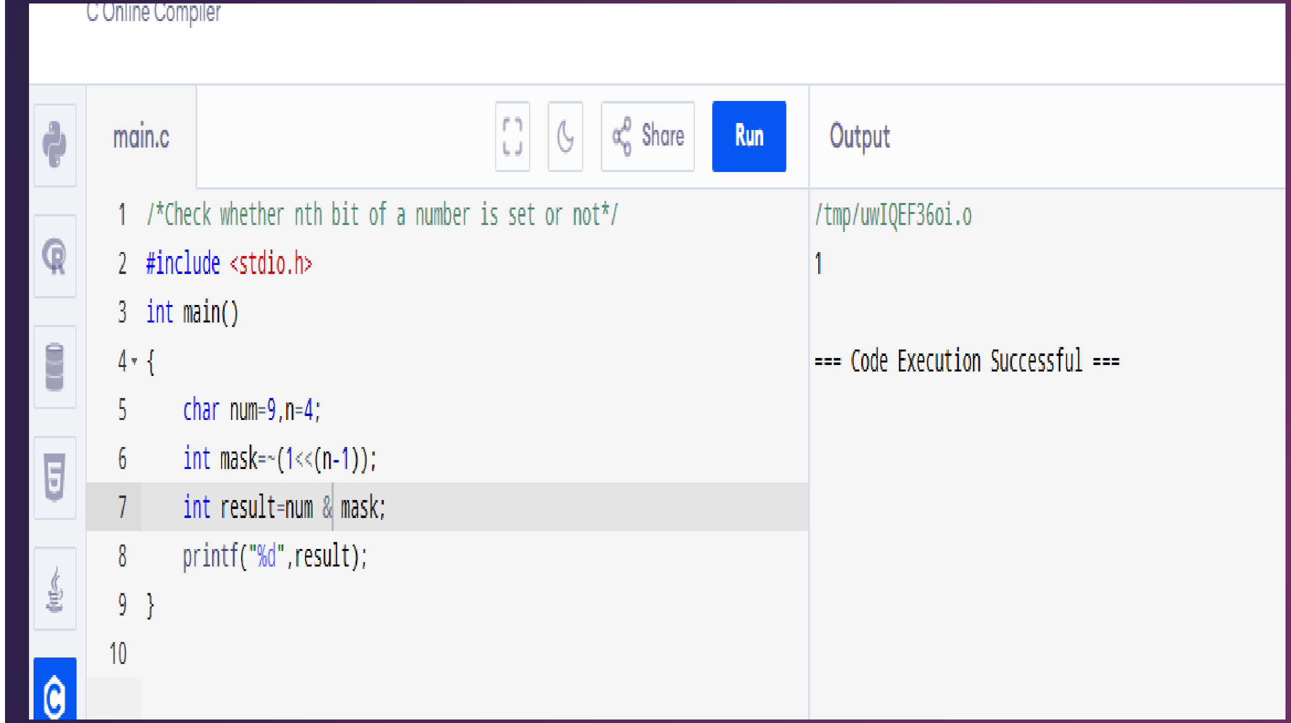
Mask :00000100

Mask=1<<(n-1)→1<<(3-1)→1<<2→00000100

Result=num | mask→00001001 | 00000100→00001101→13(dec)

4. Write a program to clear nth bit of a number.

Let’s have number as 9 and clear 4th bit.



```c
/*Check whether nth bit of a number is set or not*/
#include <stdio.h>
int main()
{
    char num=9,n=4;
    int mask=~(1<<(n-1));
    int result=num & mask;
    printf("%d",result);
}
```

```
/tmp/uwIQEF36oi.o
1


=== Code Execution Successful ===
```

Explanation:

Num=9→0 0 0 0 1 0 0 1

4th 3rd 2nd 1st

Num: 00001001

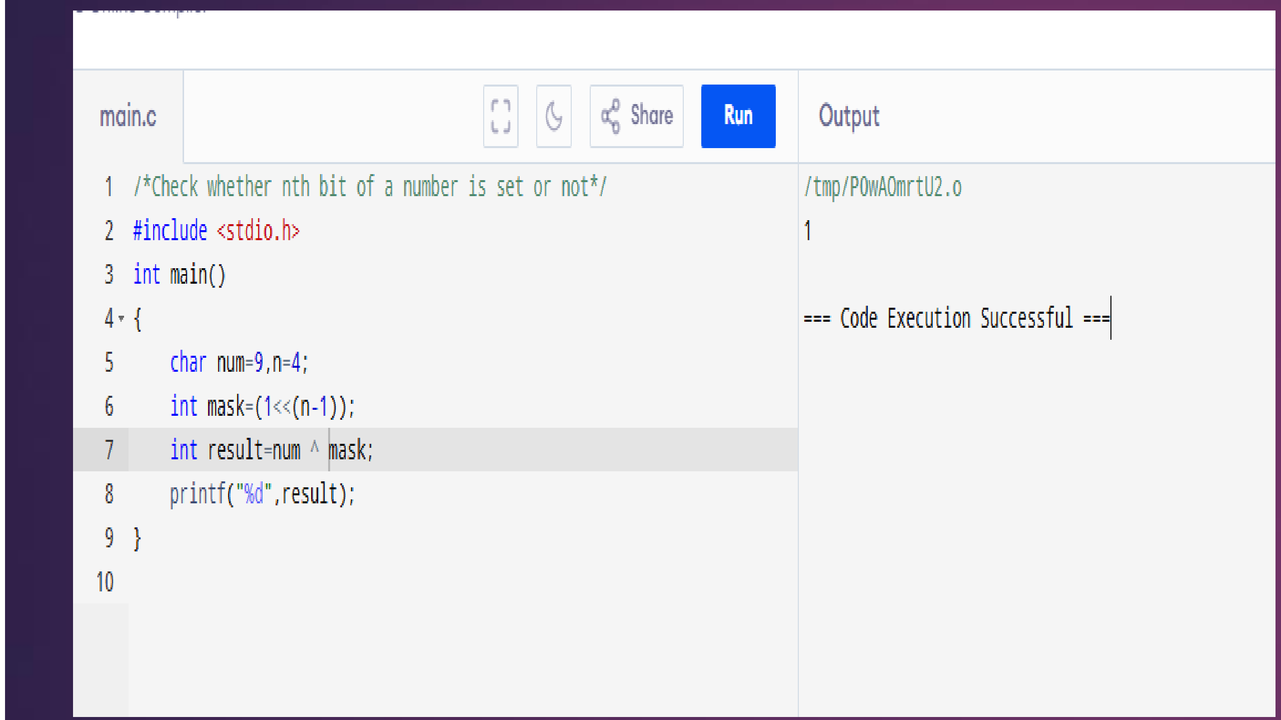
Mask :11110111

Mask=~(1<<(n-1))→~(1<<(4-1))→~(1<<3)→~(00001000)→11110111

Result=num | mask→00001001 & 11110111→00000001→1

5.Write a program to toggle nth bit of a number.

Let’s have num as 9 and clear 4th bit



```c
/*Check whether nth bit of a number is set or not*/
#include <stdio.h>
int main()
{
    char num=9,n=4;
    int mask=(1<<(n-1));
    int result=num ^ mask;
    printf("%d",result);
}

```

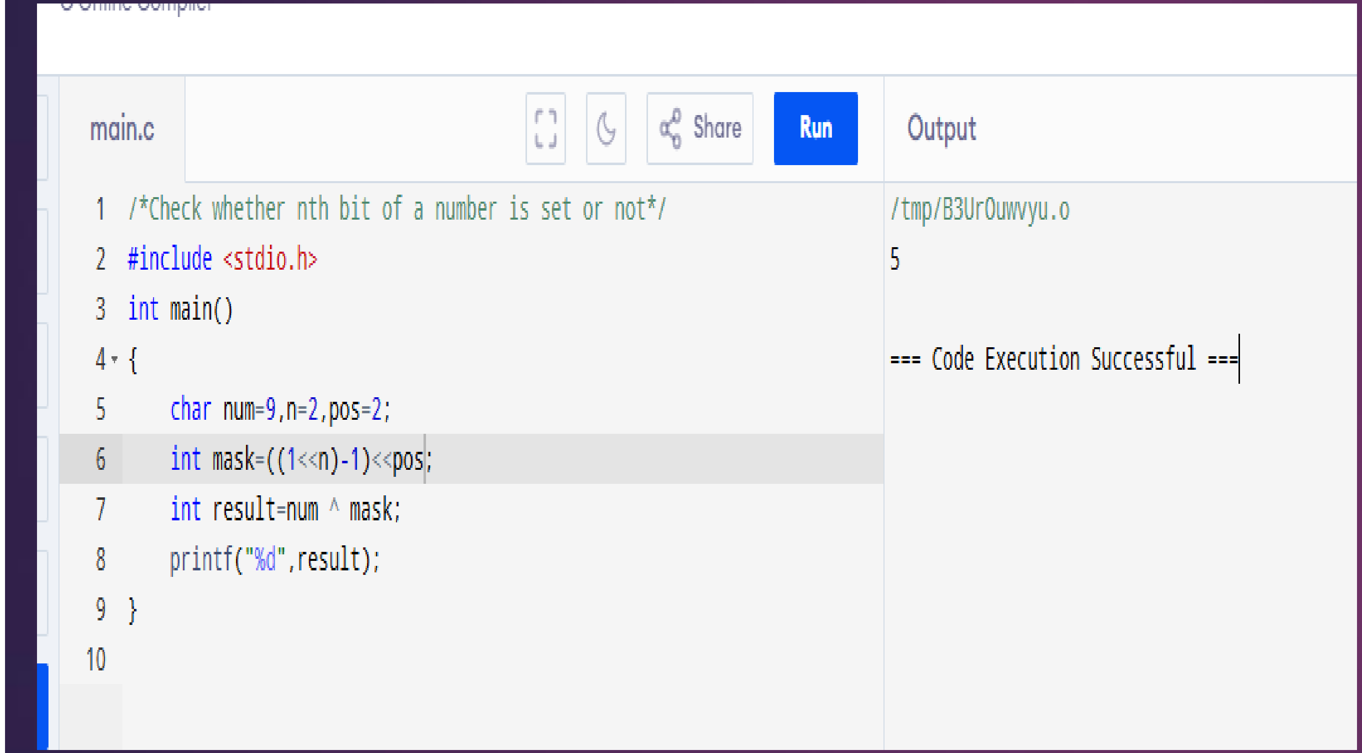
```
/tmp/POwAOmrtU2.o
1


=== Code Execution Successful ===
```

Explanation:

Num: 00001001
Mask :00001000
Mask=1<<(n-1)→1<<(2-1)→1<<1→00001000
Result=num^mask→00001001 ^ 00001000→
00000001→1(dec)

6.Write a program to toggle ‘n’ bits from given position of a number.

Let’s have num as 9, n of bits=2,pos=2



```c
/*Check whether nth bit of a number is set or not*/
#include <stdio.h>
int main()
{
    char num=9,n=2,pos=2;
    int mask=((1<<n)-1)<<pos;
    int result=num ^ mask;
    printf("%d",result);
}

```

```
/tmp/B3UrOuwvyu.o
5


=== Code Execution Successful ===
```

Explanation:

Num=9→0 0 0 0 1 0 0 1

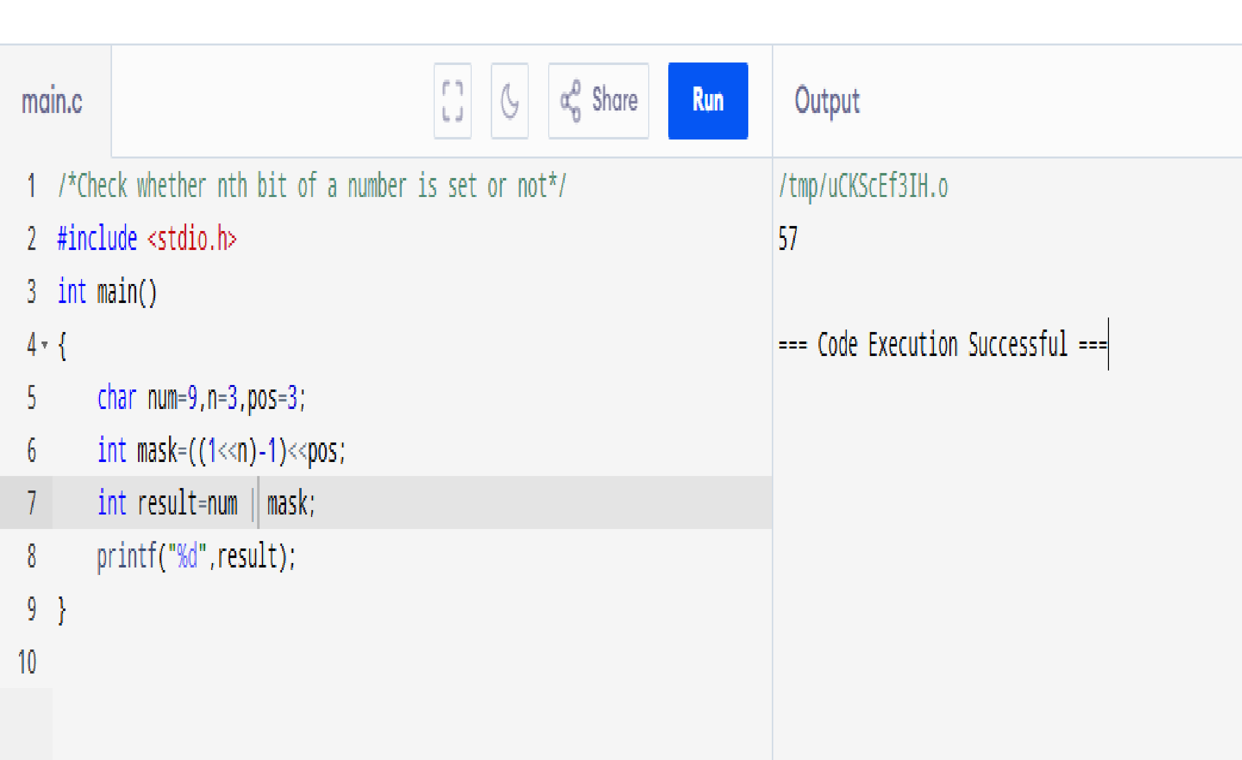
3rd 2nd 1st 0th pos

Mask= 00001100

Mask=((1<<n)-1)<<pos→((1<<2)-1)<<2)→(4-1)<<2→00001100

Result=num^mask→00001001^00001100→0000101→5

7.Write a program to set ‘n’ bits from given position of a number.
Let’s have num as 9, n of bits=3,pos=3



```c
/*Check whether nth bit of a number is set or not*/
#include <stdio.h>
int main()
{
    char num=9,n=3,pos=3;
    int mask=((1<<n)-1)<<pos;
    int result=num | mask;
    printf("%d",result);
}
```

```
/tmp/uCKScEf3IH.o
57


=== Code Execution Successful ===
```

Explanation:

Num=9→0 0 0 0 1 0 0 1

3rd 2nd 1st 0th pos

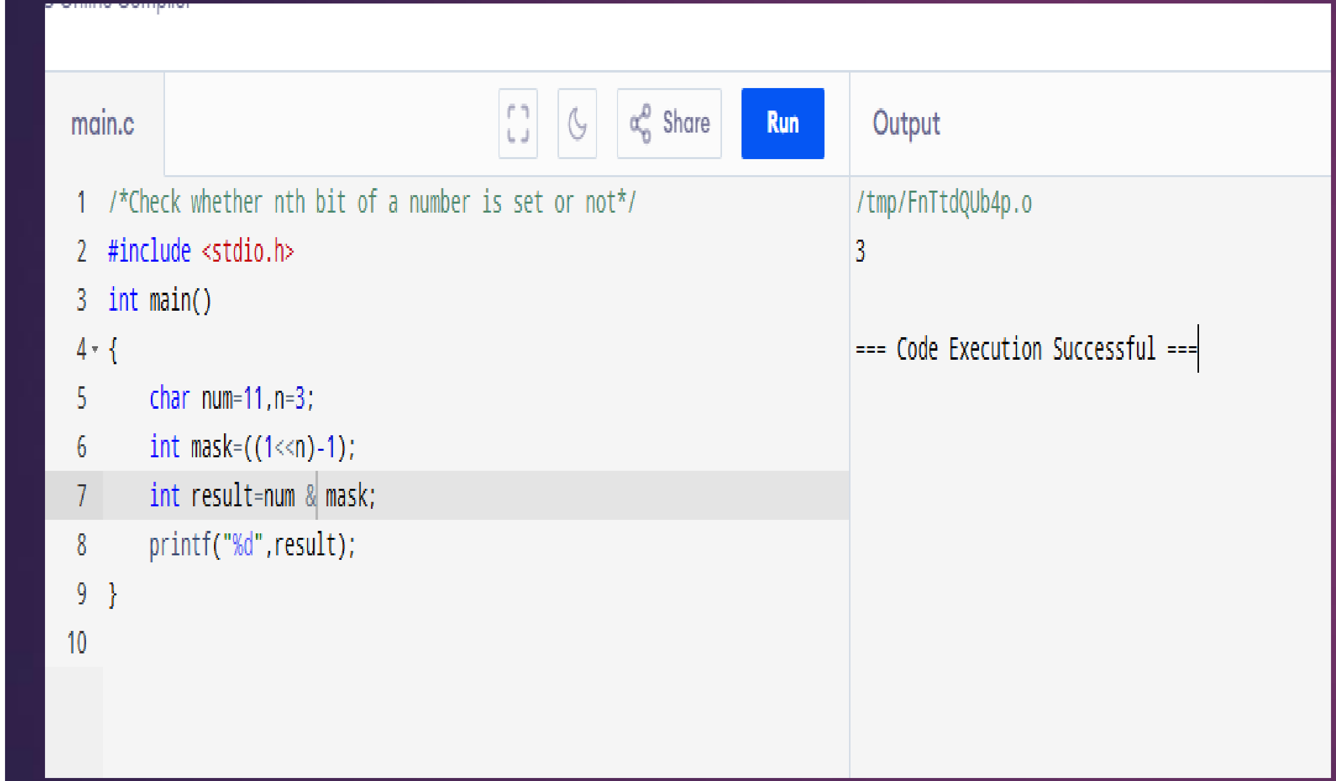
Mask= 000111000

Mask=((1<<n)-1)<<pos→((1<<3)-1)<<3)→(8-1)<<3→000111000

Result=num^mask→00001001 | 00111000→00111001→57(dec)

8. Write a program to get ‘n’ bits a of number from LSB

Let’s have num as 11, n=3



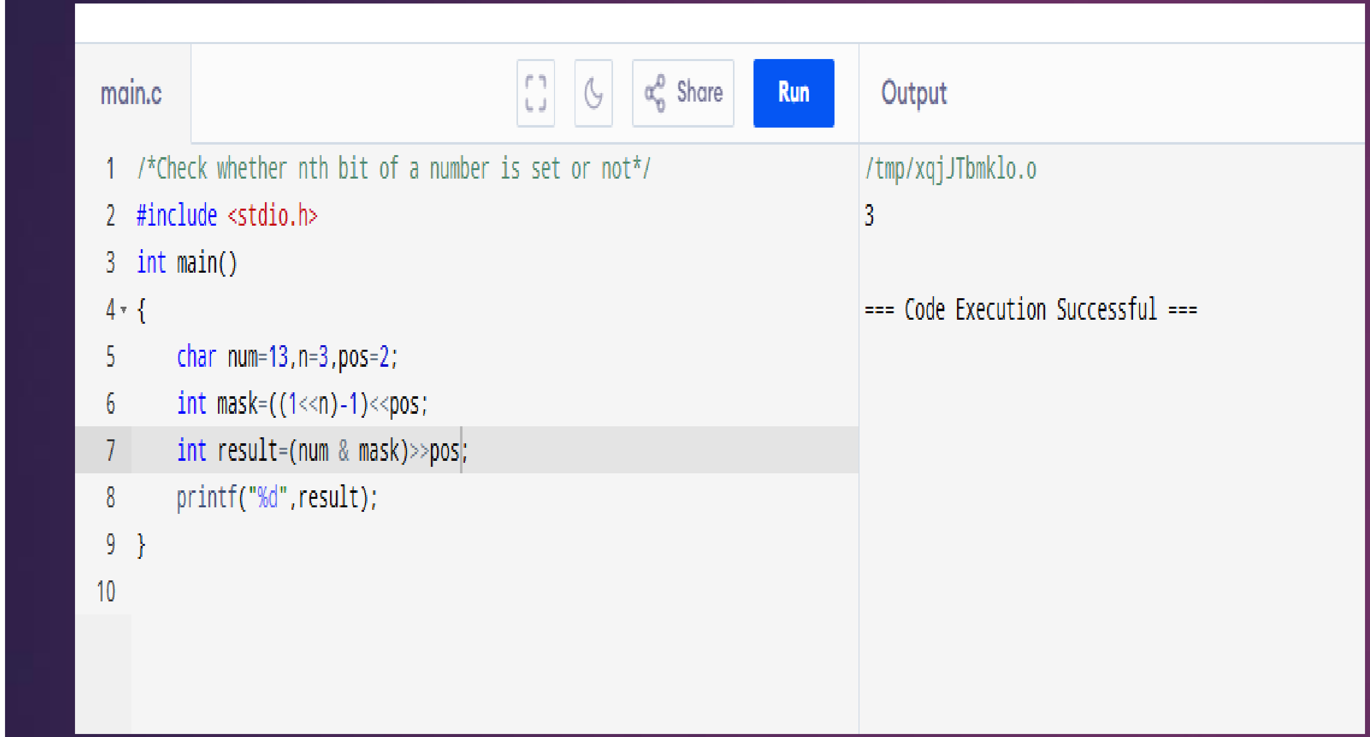
```c
/*Check whether nth bit of a number is set or not*/
#include <stdio.h>
int main()
{
    char num=11,n=3;
    int mask=((1<<n)-1);
    int result=num & mask;
    printf("%d",result);
}

```

```
/tmp/FnTtdQUb4p.o
3


=== Code Execution Successful ===
```

Explanation:

Num= 00001011
Mask=00000111
Mask=((1<<n)-1)→(1<<3)-1→8-1→7→00000111
Result=num & mask→00001011 & 00000111→00000011→3

9. Write a program to get 'n' bits from given position of a number.
Let's have num as 13, n=3 and pos=2



```c
/*Check whether nth bit of a number is set or not*/
#include <stdio.h>
int main()
{
    char num=13,n=3,pos=2;
    int mask=((1<<n)-1)<<pos;
    int result=(num & mask)>>pos;
    printf("%d",result);
}
```

```
/tmp/xqjJTbmklo.o
3


=== Code Execution Successful ===
```

Explanation:

Num=12→0 0 0 0 1 1 0 1

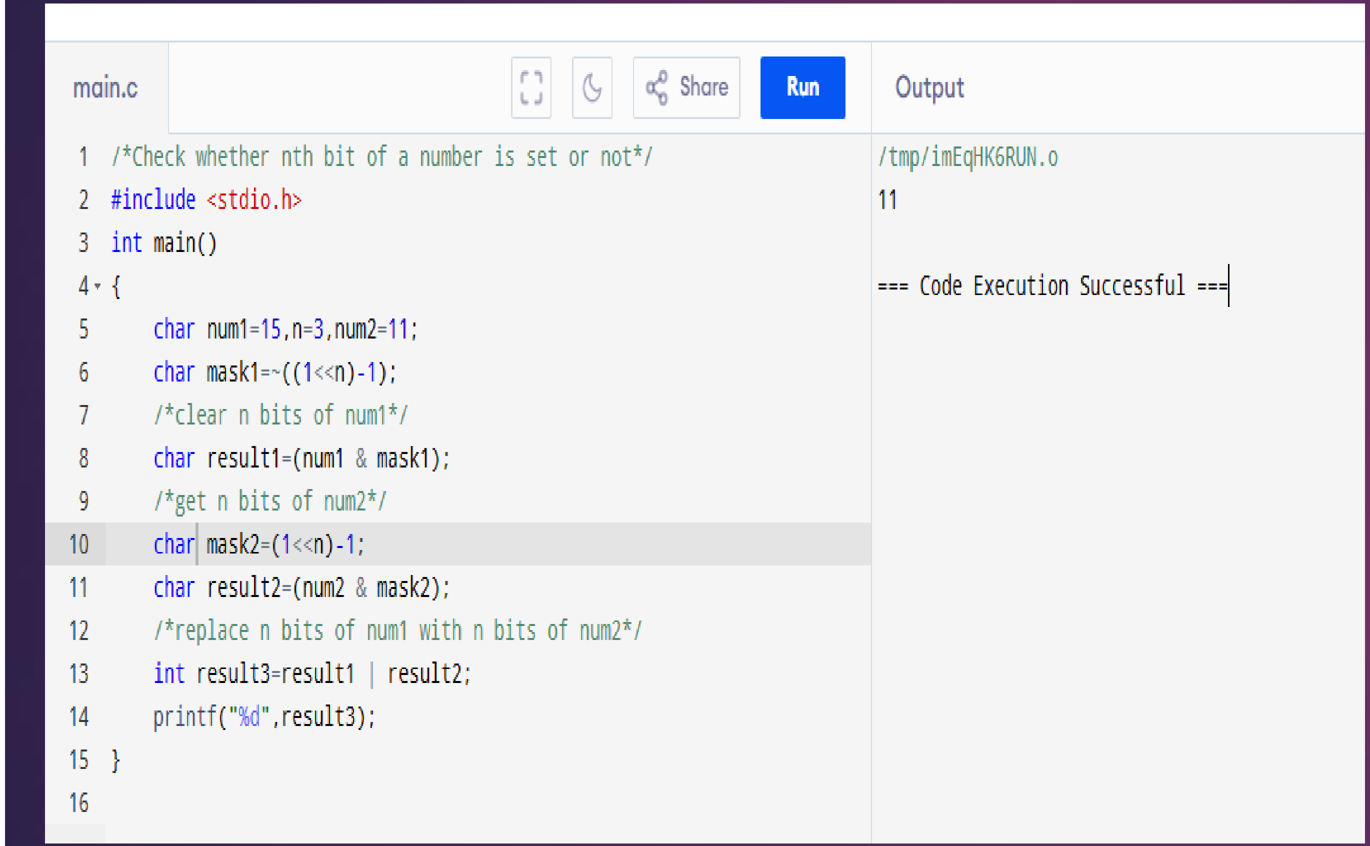
3rd 2nd 1st 0th pos

Num= 00001101

Mask=00011100

Mask=(((1<<n)-1))<<pos)→((1<<3)-1)<<2→(8-1)<<2→7<<2→00011100 Result=(num&mask)>>pos→(00001101&00011100)>>2→00001100>>2→00000011→3

10.Write a program to replace ‘n’ bits of given number num1 with ‘n’ bits of num2. Let’s have num1=15 ,n=3,num2=11;



```c
/*Check whether nth bit of a number is set or not*/
#include <stdio.h>
int main()
{
    char num1=15,n=3,num2=11;
    char mask1=~((1<<n)-1);
    /*clear n bits of num1*/
    char result1=(num1 & mask1);
    /*get n bits of num2*/
    char mask2=(1<<n)-1;
    char result2=(num2 & mask2);
    /*replace n bits of num1 with n bits of num2*/
    int result3=result1 | result2;
    printf("%d",result3);
}
```

```
/tmp/imEqHK6RUN.o
11

=== Code Execution Successful ===
```

Step1:clear n bits of num1

Num1= 00001111

Mask1=11111000

Mask1=~((1<<n)-1)→~((1<<3)-1)→~(8-1)→11111000

Result1=num1&mask1→00001111&11111000→00001000

Step2:Extract n bits from num2

Num2= 00001011

Mask2=00000111

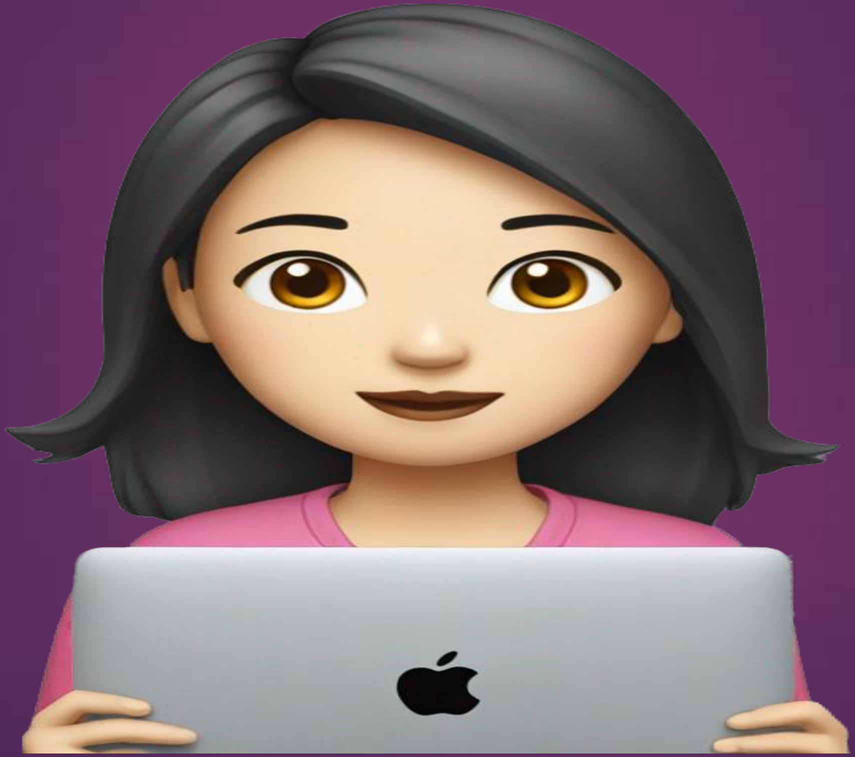
Mask2=((1<<n)-1)→(1<<3)-1→00000111

Result2=num2&mask2→00001011& 00000111→00000011

Step3:replace n bits of num1 with n bits of num2

Result3=result1|result2→00001000|00000011→00001011→11

Stay Tuned for the Level-2 Programs



Shivaleela Hoonalli