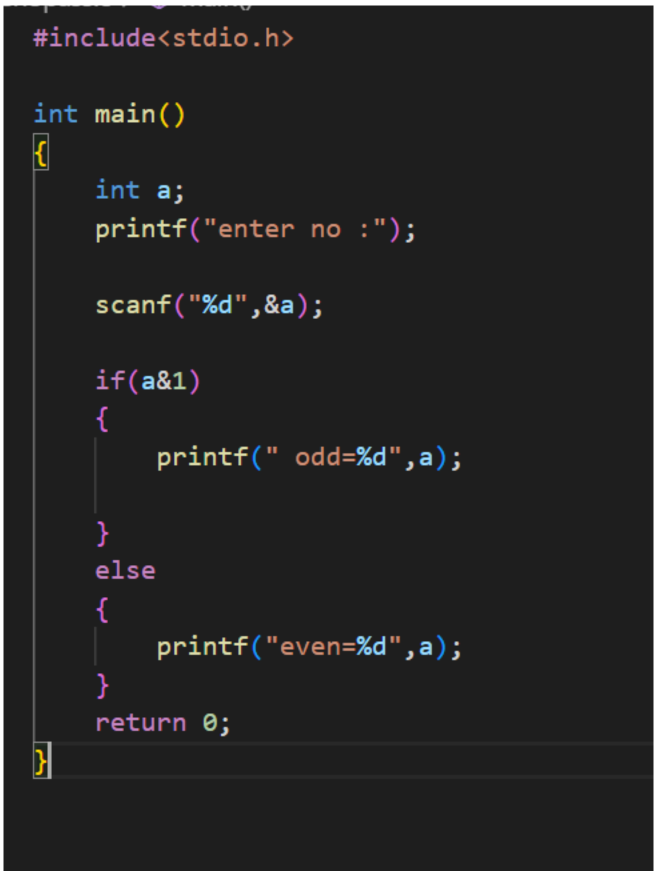
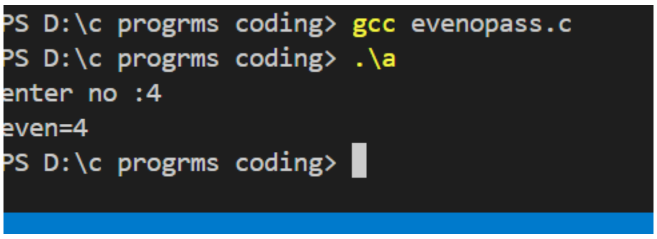
ANSU MARIUM SHIBU

DAY 4-ASSIGNMENTS

1. Write a C program to determine if the least significant bit of a given integer is set (i.e., check if the number is odd).

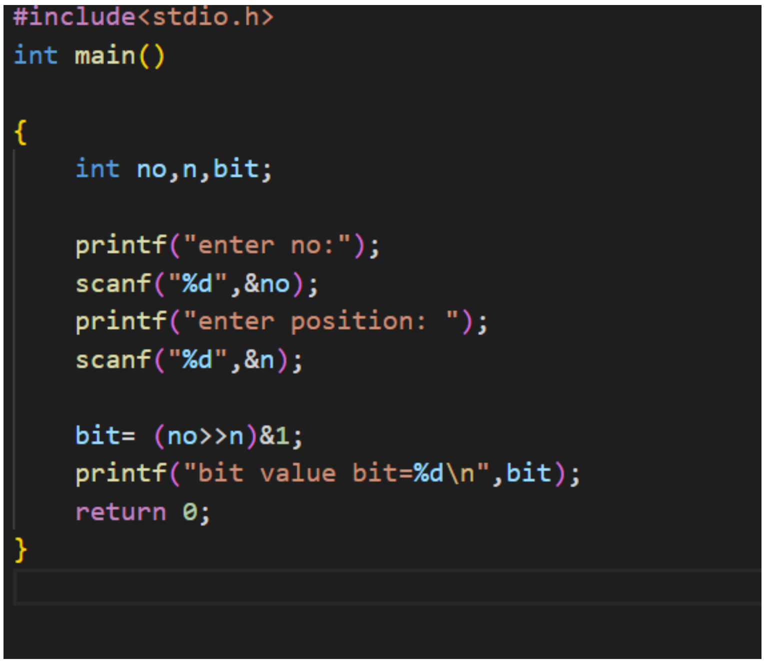


```c
#include<stdio.h>

int main()
{
    int a;
    printf("enter no :");

    scanf("%d",&a);

    if(a&1)
    {
        printf(" odd=%d",a);

    }
    else
    {
        printf("even=%d",a);
    }
    return 0;
}
```

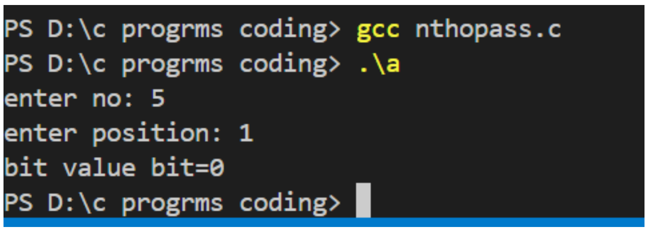


```
PS D:\c progrms coding> gcc evenopass.c
PS D:\c progrms coding> .\a
enter no :4
even=4
PS D:\c progrms coding> 
```

2.Create a C program that retrieves the value of the nth bit from a given integer.

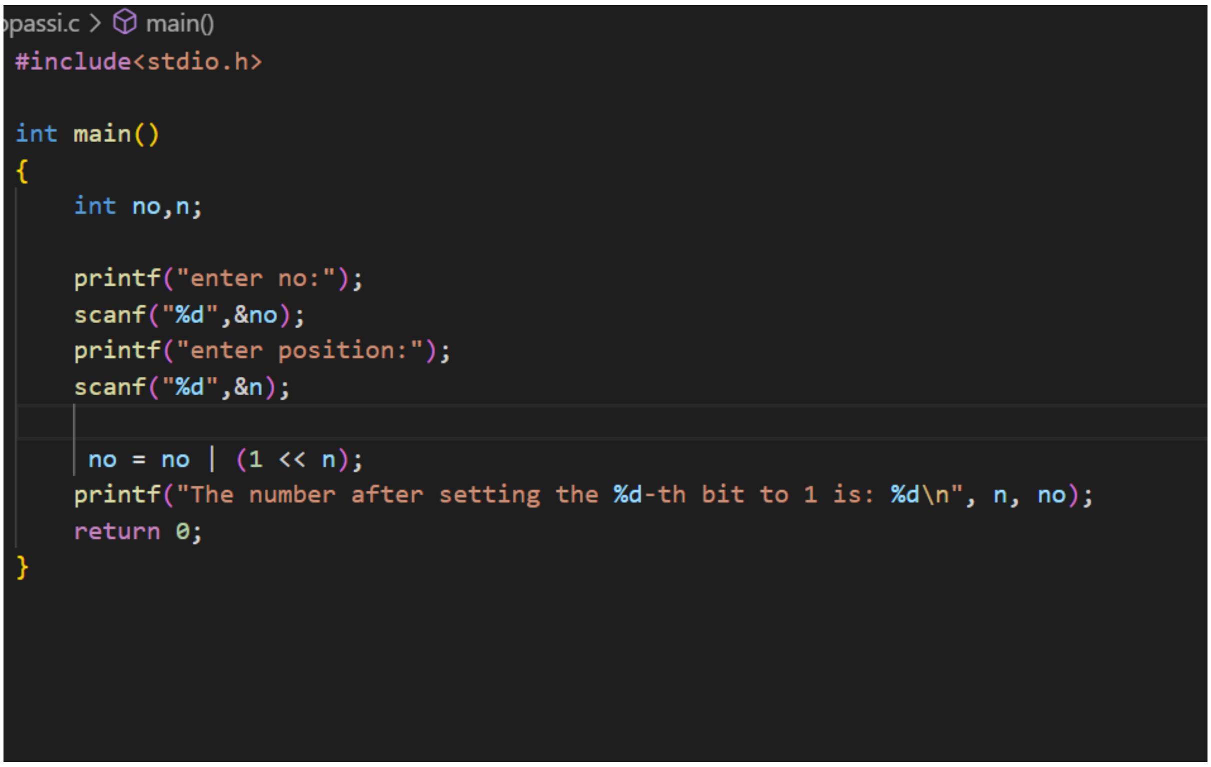


```c
#include<stdio.h>
int main()

{
    int no,n,bit;

    printf("enter no:");
    scanf("%d",&no);
    printf("enter position: ");
    scanf("%d",&n);

    bit= (no>>n)&1;
    printf("bit value bit=%d\n",bit);
    return 0;
}
```

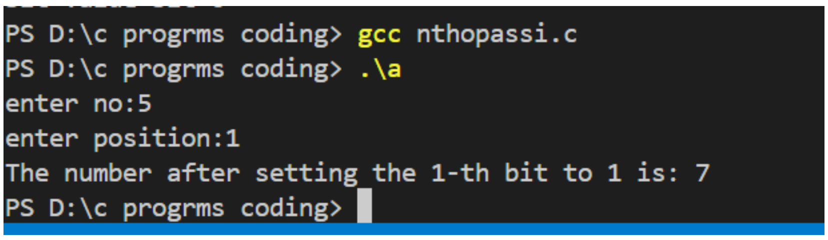


```
PS D:\c progrms coding> gcc nthopass.c
PS D:\c progrms coding> .\a
enter no: 5
enter position: 1
bit value bit=0
PS D:\c progrms coding> 
```

3.Develop a C program that sets the nth bit of a given integer to 1.

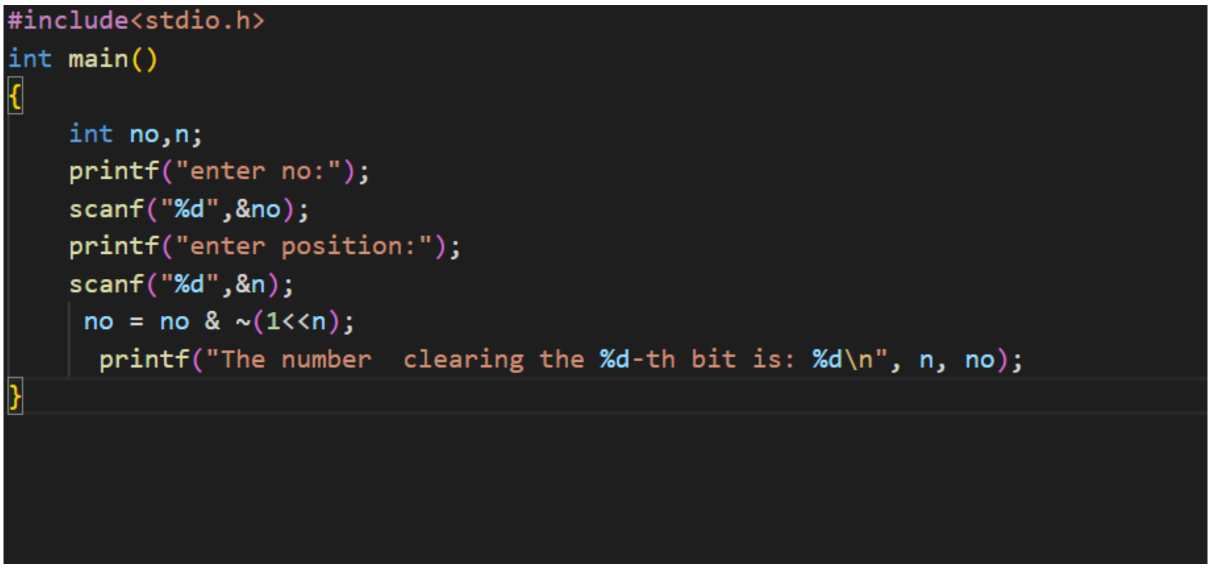


```c
passi.c > main()
#include<stdio.h>

int main()
{
    int no,n;

    printf("enter no:");
    scanf("%d",&no);
    printf("enter position:");
    scanf("%d",&n);

     no = no | (1 << n);
    printf("The number after setting the %d-th bit to 1 is: %d\n", n, no);
    return 0;
}
```

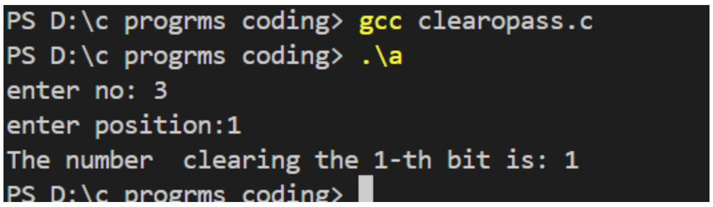


```
PS D:\c progrms coding> gcc nthopassi.c
PS D:\c progrms coding> .\a
enter no:5
enter position:1
The number after setting the 1-th bit to 1 is: 7
PS D:\c progrms coding>
```

4.Write a C program that clears (sets to 0) the nth bit of a given integer.

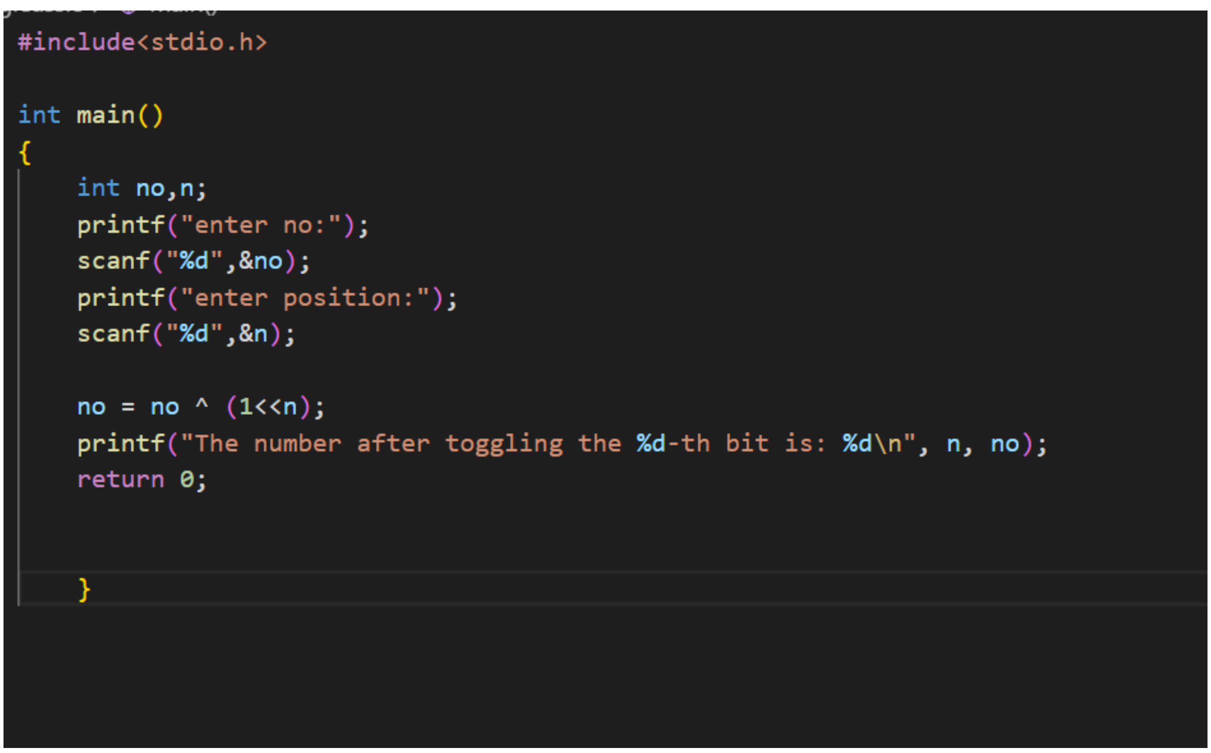


```c
#include<stdio.h>
int main()
{
    int no,n;
    printf("enter no:");
    scanf("%d",&no);
    printf("enter position:");
    scanf("%d",&n);
     no = no & ~(1<<n);
      printf("The number  clearing the %d-th bit is: %d\n", n, no);
}
```

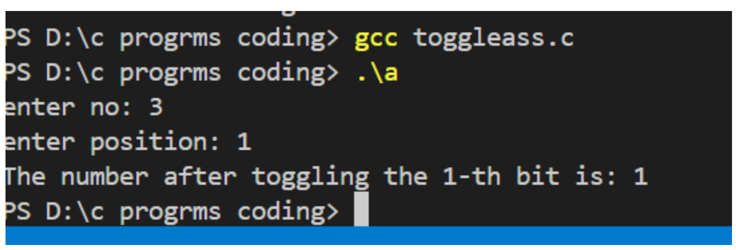


```
PS D:\c progrms coding> gcc clearopass.c
PS D:\c progrms coding> .\a
enter no: 3
enter position:1
The number  clearing the 1-th bit is: 1
PS D:\c progrms coding>
```

5.Create a C program that toggles the nth bit of a given integer.



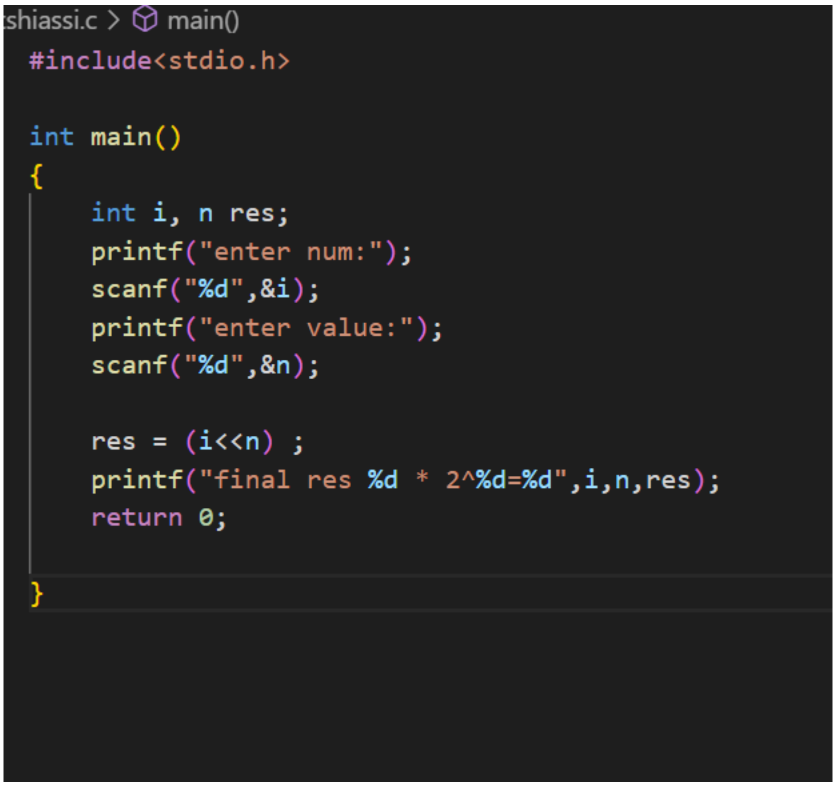
```c
#include<stdio.h>

int main()
{
    int no,n;
    printf("enter no:");
    scanf("%d",&no);
    printf("enter position:");
    scanf("%d",&n);

    no = no ^ (1<<n);
    printf("The number after toggling the %d-th bit is: %d\n", n, no);
    return 0;


    }
```



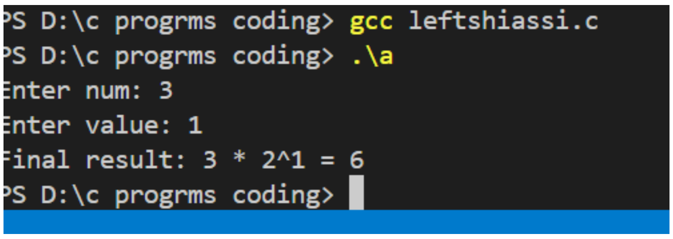
```
PS D:\c progrms coding> gcc toggleass.c
PS D:\c progrms coding> .\a
enter no: 3
enter position: 1
The number after toggling the 1-th bit is: 1
PS D:\c progrms coding>
```

6. Write a C program that takes an integer input and multiplies it by

2^n using the left shift operator.

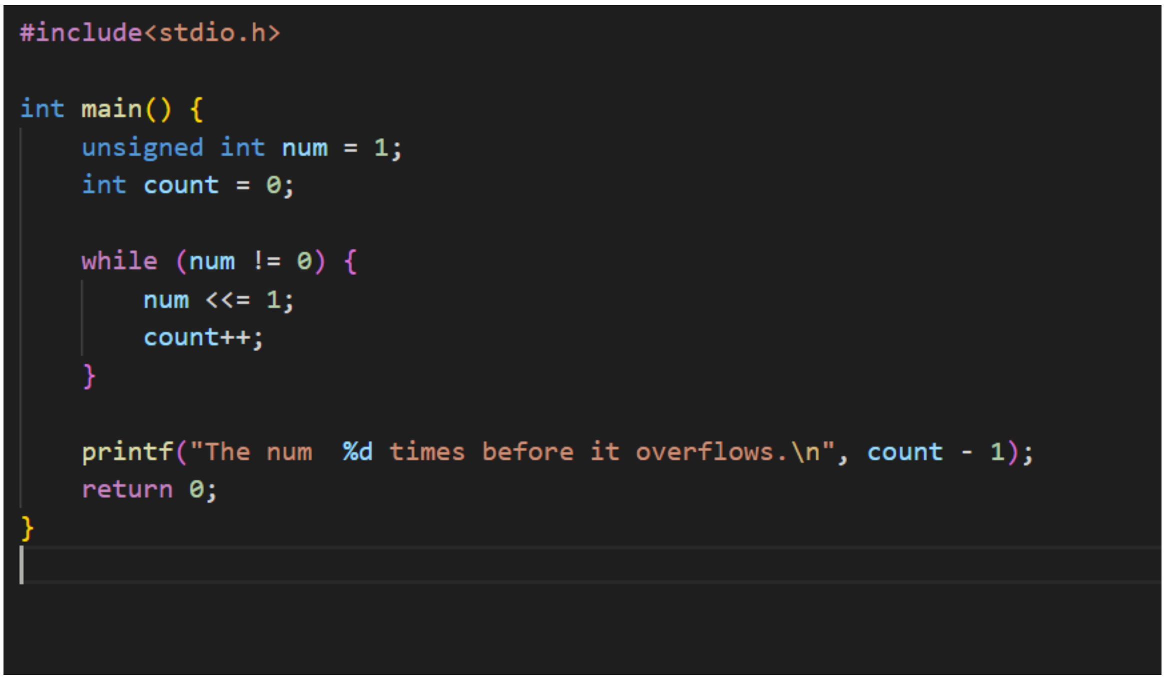


```
shiassi.c > main()
#include<stdio.h>

int main()
{
    int i, n res;
    printf("enter num:");
    scanf("%d",&i);
    printf("enter value:");
    scanf("%d",&n);

    res = (i<<n) ;
    printf("final res %d * 2^%d=%d",i,n,res);
    return 0;

}
```

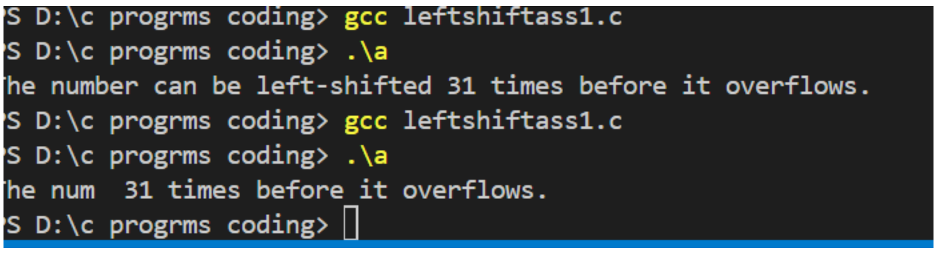


```
PS D:\c progrms coding> gcc leftshiassi.c
PS D:\c progrms coding> .\a
Enter num: 3
Enter value: 1
Final result: 3 * 2^1 = 6
PS D:\c progrms coding>
```

7..Create a C program that counts how many times you can left shift a number before it overflows (exceeds the maximum value for an integer).

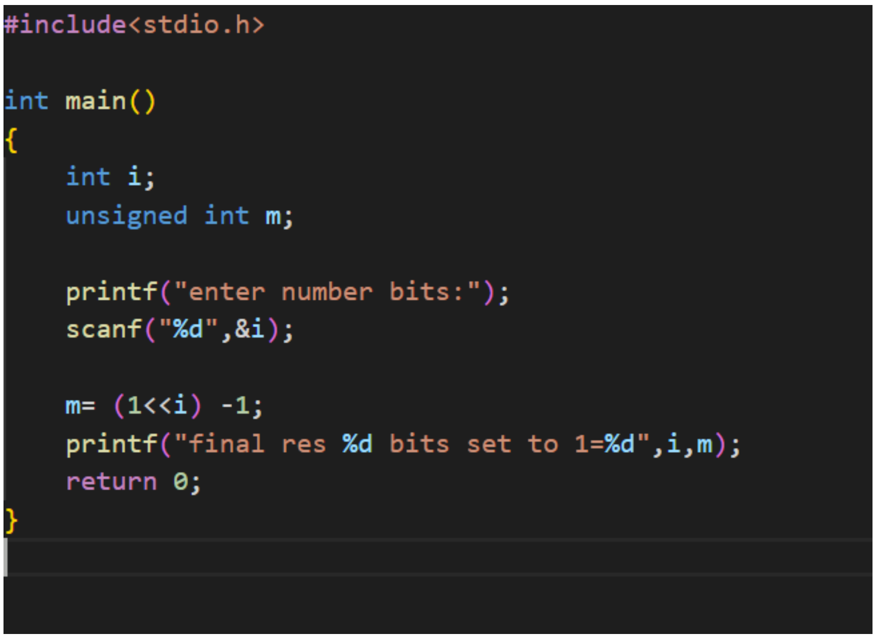


```c
#include<stdio.h>

int main() {
    unsigned int num = 1;
    int count = 0;

    while (num != 0) {
        num <<= 1;
        count++;
    }

    printf("The num  %d times before it overflows.\n", count - 1);
    return 0;
}
```

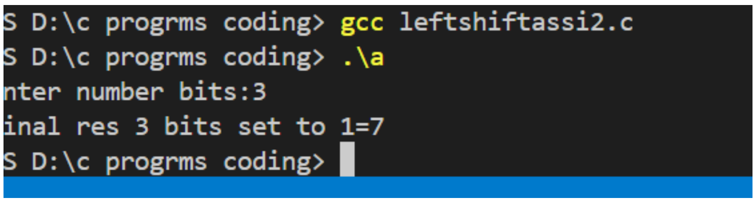


```
S D:\c progrms coding> gcc leftshiftass1.c
S D:\c progrms coding> .\a
he number can be left-shifted 31 times before it overflows.
S D:\c progrms coding> gcc leftshiftass1.c
S D:\c progrms coding> .\a
he num  31 times before it overflows.
S D:\c progrms coding>
```

8. Write a C program that creates a bitmask with the first n bits set to 1 using the left shift operator.

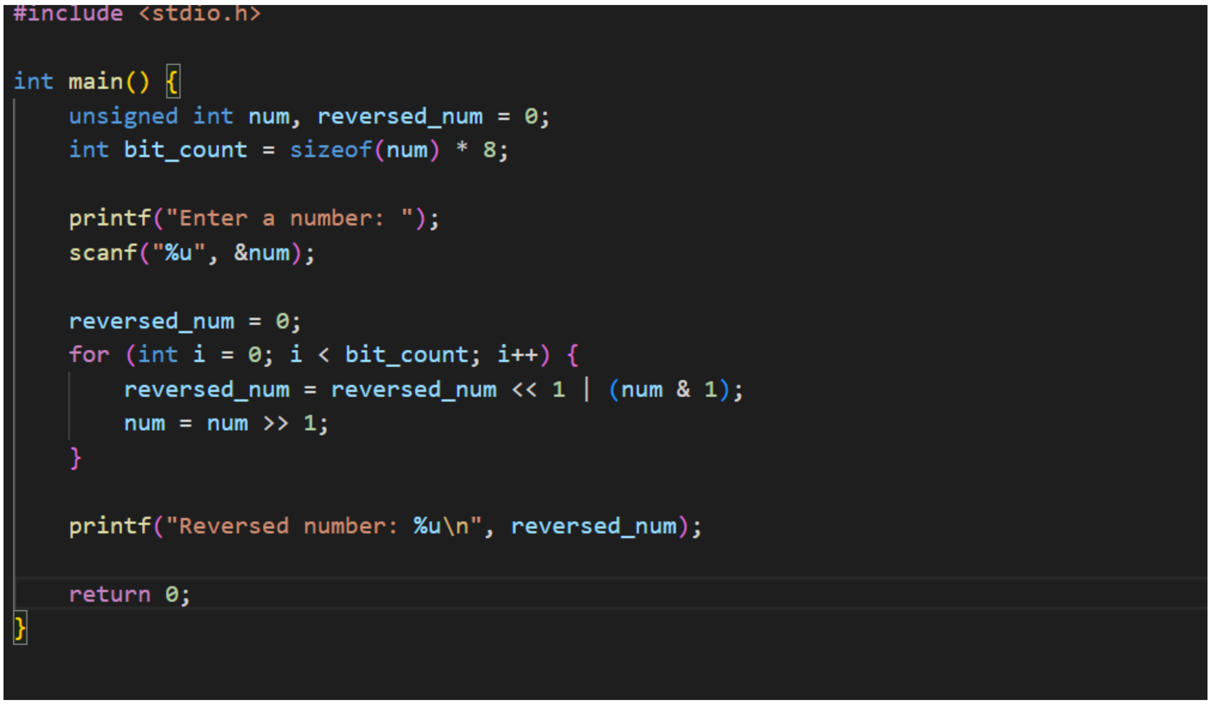


```c
#include<stdio.h>

int main()
{
    int i;
    unsigned int m;

    printf("enter number bits:");
    scanf("%d",&i);

    m= (1<<i) -1;
    printf("final res %d bits set to 1=%d",i,m);
    return 0;
}
```

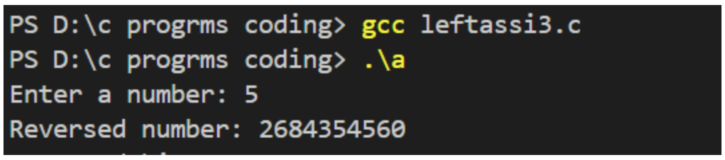


```
S D:\c progrms coding> gcc leftshiftassi2.c
S D:\c progrms coding> .\a
nter number bits:3
inal res 3 bits set to 1=7
S D:\c progrms coding>
```

9. Develop a C program that reverses the bits of an integer using left shift and right shift operations.

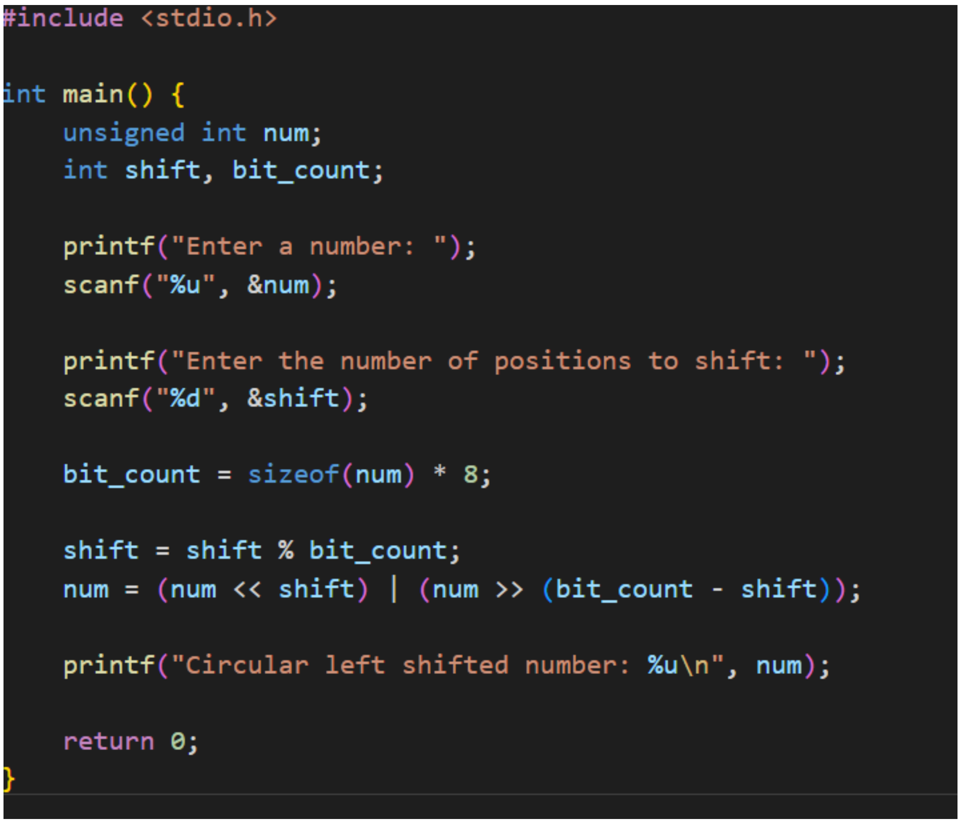


```c
#include <stdio.h>

int main() {
    unsigned int num, reversed_num = 0;
    int bit_count = sizeof(num) * 8;

    printf("Enter a number: ");
    scanf("%u", &num);

    reversed_num = 0;
    for (int i = 0; i < bit_count; i++) {
        reversed_num = reversed_num << 1 | (num & 1);
        num = num >> 1;
    }

    printf("Reversed number: %u\n", reversed_num);

    return 0;
}
```

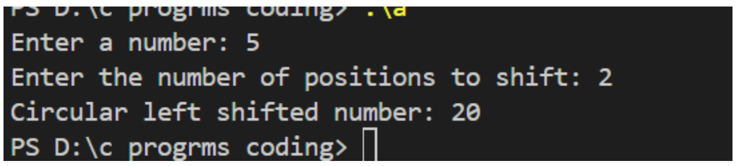


```
PS D:\c progrms coding> gcc leftassi3.c
PS D:\c progrms coding> .\a
Enter a number: 5
Reversed number: 2684354560
```

10. Create a C program that performs a circular left shift on an integer.

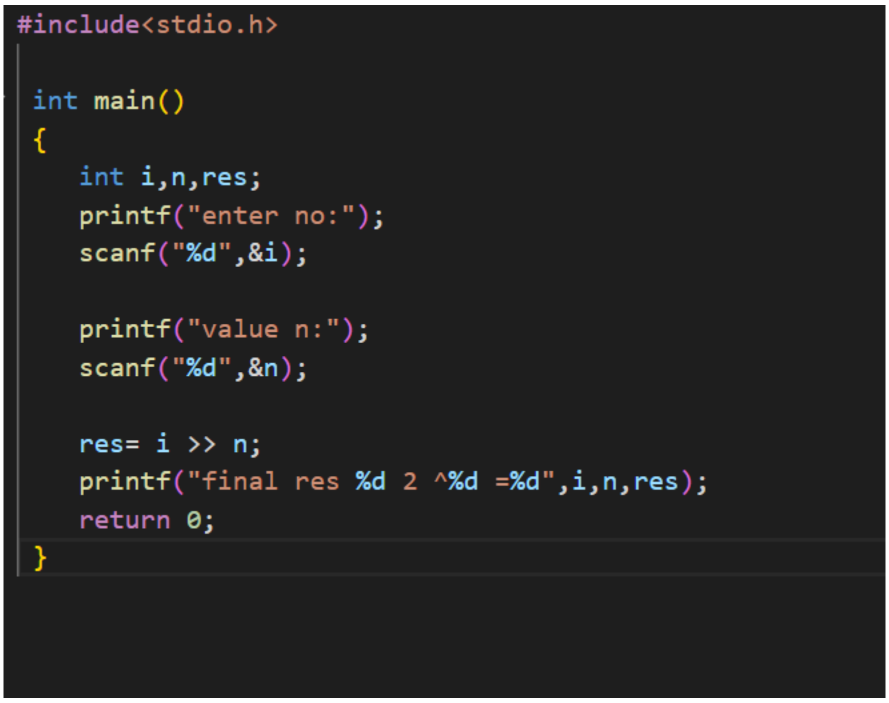


```c
#include <stdio.h>

int main() {
    unsigned int num;
    int shift, bit_count;

    printf("Enter a number: ");
    scanf("%u", &num);

    printf("Enter the number of positions to shift: ");
    scanf("%d", &shift);

    bit_count = sizeof(num) * 8;

    shift = shift % bit_count;
    num = (num << shift) | (num >> (bit_count - shift));

    printf("Circular left shifted number: %u\n", num);

    return 0;
}
```

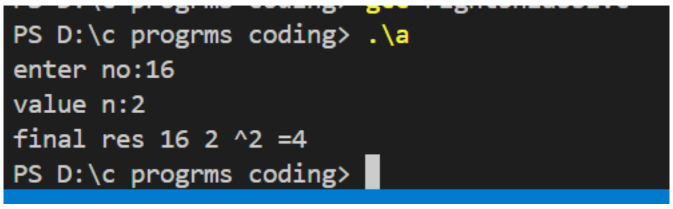


```
PS D:\c progrms coding> .\a
Enter a number: 5
Enter the number of positions to shift: 2
Circular left shifted number: 20
PS D:\c progrms coding>
```

11. Write a C program that takes an integer input and divides it by 2^ n using the right shift operator.

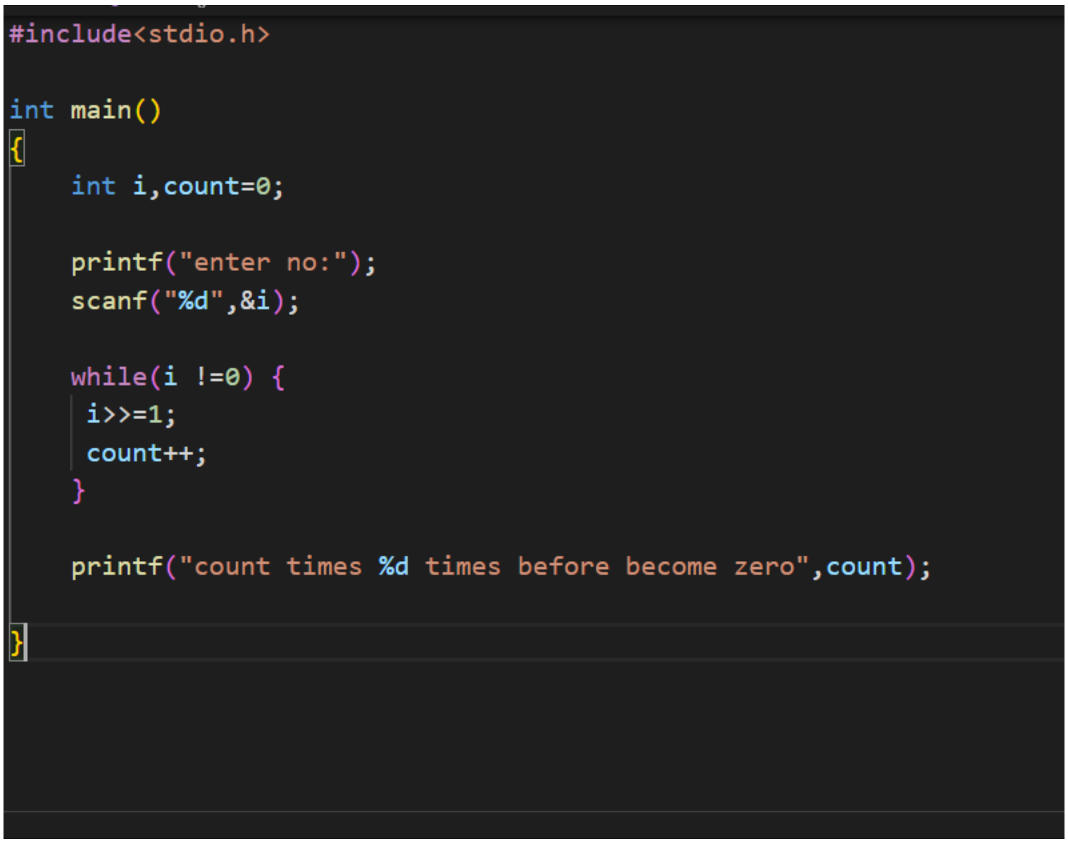


```c
#include<stdio.h>

int main()
{
    int i,n,res;
    printf("enter no:");
    scanf("%d",&i);

    printf("value n:");
    scanf("%d",&n);

    res= i >> n;
    printf("final res %d 2 ^%d =%d",i,n,res);
    return 0;
}
```

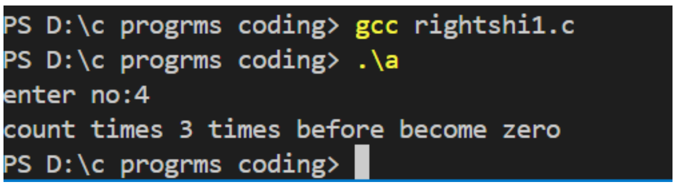


```
PS D:\c progrms coding> .\a
enter no:16
value n:2
final res 16 2 ^2 =4
PS D:\c progrms coding>
```

12. Create a C program that counts how many times you can right shift a number before it becomes zero.

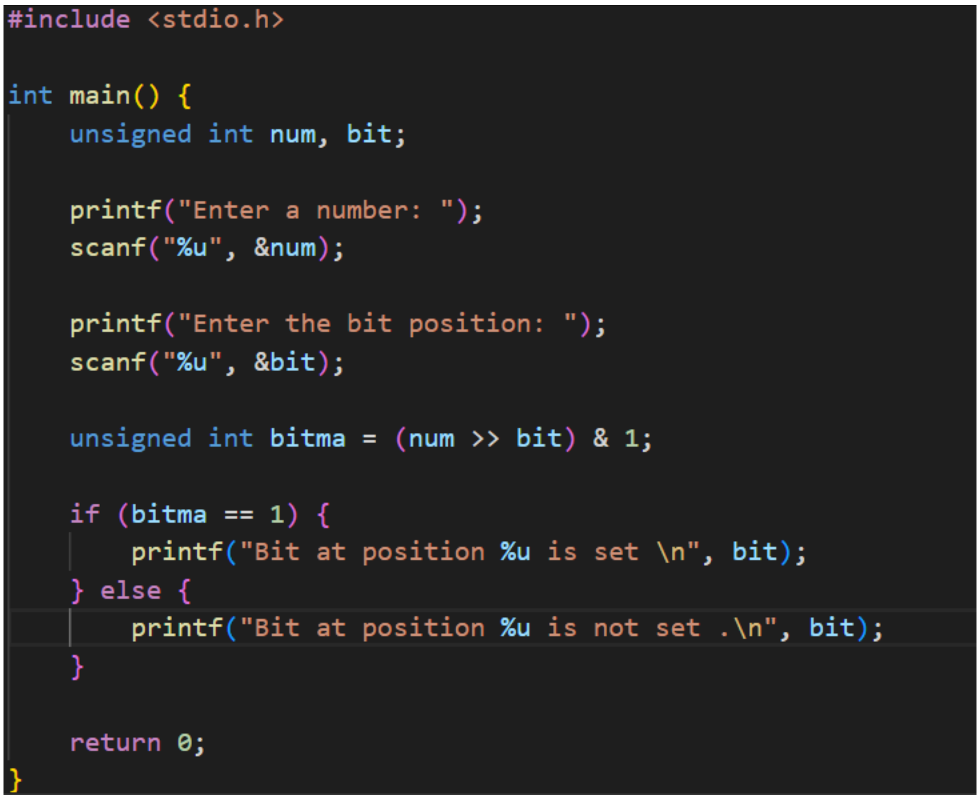


```c
#include<stdio.h>

int main()
{
    int i,count=0;

    printf("enter no:");
    scanf("%d",&i);

    while(i !=0) {
     i>>=1;
     count++;
    }

    printf("count times %d times before become zero",count);

}
```

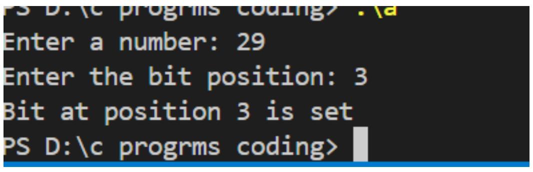


```
PS D:\c progrms coding> gcc rightshi1.c
PS D:\c progrms coding> .\a
enter no:4
count times 3 times before become zero
PS D:\c progrms coding> 
```

13. Develop a C program that uses the right shift operator to create a bitmask that checks if specific bits are set in an integer.

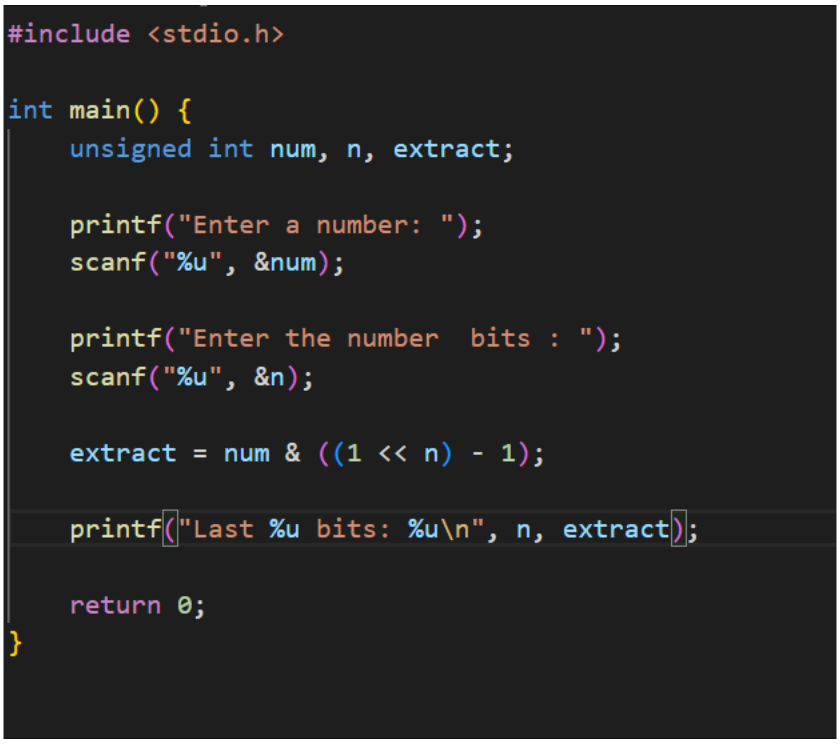


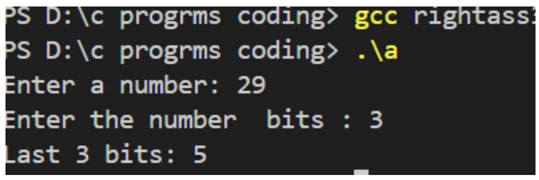
```c
#include <stdio.h>

int main() {
    unsigned int num, bit;

    printf("Enter a number: ");
    scanf("%u", &num);

    printf("Enter the bit position: ");
    scanf("%u", &bit);

    unsigned int bitma = (num >> bit) & 1;

    if (bitma == 1) {
        printf("Bit at position %u is set \n", bit);
    } else {
        printf("Bit at position %u is not set .\n", bit);
    }

    return 0;
}
```



```
PS D:\c progrms coding> .\a
Enter a number: 29
Enter the bit position: 3
Bit at position 3 is set
PS D:\c progrms coding>
```

14. Write a C program that extracts the last n bits from a given integer using the right shift operator.


```c
#include <stdio.h>

int main() {
    unsigned int num, n, extract;

    printf("Enter a number: ");
    scanf("%u", &num);

    printf("Enter the number  bits : ");
    scanf("%u", &n);

    extract = num & ((1 << n) - 1);

    printf("Last %u bits: %u\n", n, extract);

    return 0;
}
```


```
PS D:\c progrms coding> gcc rightass
PS D:\c progrms coding> .\a
Enter a number: 29
Enter the number  bits : 3
Last 3 bits: 5
```