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 a number\n");
    scanf("%d",&a);
    if(a & 1)
    {
        printf("ODD Number");
    }
    else
    {
        printf("EVEN Number");
    }
    return 0;
}
```

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

```c
#include<stdio.h>
int binary(int num,int n){
    int i,bit,b[32];
    for(i=0;i<32;i++){
        b[i]=num%2;
        num/=2;
    }
    bit=b[n];
    return bit;
}
int main(){
    int num,n,bit;
    printf("Enter a Number\n");
    scanf("%d",&num);
    printf("Enter value of n::\n");
    scanf("%d",&n);
    bit=binary(num,n);
    printf("nth bit is::%d\n",bit);

    return 0;
}
```

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

```c
#include<stdio.h>
int main(){
    int num, n,res;
    printf("Enter an integer: ");
    scanf("%d", &num);
    printf("Enter the bit position to set:\n ");
    scanf("%d", &n);
    res= (num | (1 << n));
    printf("New value is %d\n",res);
    return 0;
 }
```

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

```c
#include<stdio.h>
int main(){
    int num, n,res;
    printf("Enter an integer: ");
    scanf("%d", &num);
    printf("Enter the bit position to clear:\n ");
    scanf("%d", &n);
    res= (num & ~(1 << n));
    printf("New value is %d\n",res);
    return 0;
 }
```

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

```c
#include<stdio.h>
int main(){
    int num, n,res;
    printf("Enter an integer: ");
    scanf("%d", &num);
    printf("Enter the bit position to toggle:\n ");
    scanf("%d", &n);
    res= (num ^ (1 << n));
    printf("New value is %d\n",res);
    return 0;
 }
```

1. Write a C program that takes an integer input and multiplies it by 2^n using the left shift operator.

```c
#include<stdio.h>
int main(){
    int num,n,p;
    printf("Enter a number::\n");
    scanf("%d",&num);
    printf("Enter power of 2^::\n");
    scanf("%d",&n);
    p=num << n;
    printf("Result::%d\n",p);
    return 0;
}
```

2. 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(){
    int i=1,count=0,p,num=1;
    printf("Running\n");
    while(i!=0){
        p=num << i;
        count+=1;
        i+=i;

    }
    printf("Count::%d",count);
    return 0;
}
```

3. 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 n,mask;
    printf("Enter value of n\n");
```

```
    scanf("%d",&n);
    mask=(1<<n);
    printf("MASK::%d\n",mask);
    return 0;

}
```

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

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

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

```
#include<stdio.h>
int main(){
    int num,n,p;
    printf("Enter a number::\n");
    scanf("%d",&num);
    printf("Enter power of 2^::\n");
    scanf("%d",&n);
    p=num >> n;
    printf("Result::%d\n",p);
    return 0;
}
```

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

```
#include<stdio.h>
int main(){
    int i=1,count=0,p,num=1;
    printf("Running\n");
    while(i!=0){
        p=num >> i;
        count+=1;
        i+=i;

    }
    printf("Count::%d",count);
    return 0;
}
```

3. 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(){
    int num,n,res;
    printf("Enter an integer: ");
    scanf("%d", &num);
    printf("Enter value of last bit to extract :\n ");
    scanf("%d", &n);
    res=num >> (32 - n);
    printf("Set value is %d\n",res);
 }
```

4. 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(){
    int num, pos,res;
    printf("Enter an integer: ");
    scanf("%d", &num);
    printf("Enter the bit position to check:\n ");
    scanf("%d", &pos);
    res=(num >> pos) & 1;
    printf("Set value is %d\n",res);
 }
```