

Class assignment 1

```
#include <stdio.h>
void myfun(void);

int main()
{
    myfun();
    myfun();
    myfun();
    myfun();

    return 0;
}
void myfun(void)
{
    static int count = 0;
    count = count + 1;
    printf("The function is executed %d times\n",count);
}
```

Class Assignment 2

```
Main.c
#include <stdio.h>
void TestFile_myfun();
int mainPrivateData;
int main()
{
    mainPrivateData = 100;
    printf("001 mainPrivateData = %d\n",mainPrivateData);

    TestFile_myfun();
    printf("001 mainPrivateData = %d\n",mainPrivateData);
    return 0;
}
```

```
TestFile.c
extern int mainPrivateData;
//extern void change_clock(int);
void TestFile_myfun(){
    mainPrivateData = 500;
    //change_clock = 500;
}
```

Class Assignment 3

Main.c

```
#include <stdio.h>
void TestFile_myfun(void);
void change_clock(int);
//int mainPrivateData;
int main()
{
    TestFile_myfun();
    return 0;

}
void change_clock(int system_clock)
{
    printf("System clock change to %d",system_clock);
}
```

TestFile.c

```
extern int mainPrivateData;
extern void change_clock(int);
void TestFile_myfun(){

    change_clock(500);
}
```

Class Assignment 4

```
#include <stdio.h>
int main()
{
    char A =40;
    char B = 30;
    printf("The output after bitwise OR(|) operation is %d\n", (A|B));
    printf("The output after bitwise AND(&) operation is %d\n", (A&B));
    printf("The output after bitwise XOR(^) operation is %d\n", (A^B));
    printf("The output after bitwise NOT(~) operation is %d\n", (~A));

}
```

Assignment 1:(Using Bitwise operators)

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).

```
#include <stdio.h>
int main()
{
    int num;
    scanf("%d",&num);
```

```

int res = num & 1;;
if(res == 0)
{
    printf("Even Number\n");
}
else
{
    printf("Odd number\n");
}
}

```

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

```

#include <stdio.h>
int main()
{
    int num;
    int n;
    scanf("%d %d",&num,&n);
    int res = num & (1 << n);
    printf("%d th bit is %d",n,res);

}

```

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

```

#include <stdio.h>
int main()
{
    int num;
    int n;
    scanf("%d %d",&num,&n);
    int res = num | (1 << n);
    printf("The result is :%d",res);

}

```

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

```

#include <stdio.h>
int main()
{
    int num;
    int n;
    scanf("%d %d",&num,&n);
    int res = num & ~(1 << n);
    printf("The result is :%d",res);

}

```

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

```
#include <stdio.h>
int main()
{
    int num;
    int n;
    scanf("%d %d",&num,&n);
    int res = num ^ (1<<n);
    printf("The result is :%d",res);

}
```

Class Assignment 5

```
#include <stdio.h>
int main()
{
    int num;

    scanf("%x",&num);
    printf("Set 4th bit :%x\n",num | (1<<4));
    printf("Set 6th bit :%x\n",num | (1<<6));
    printf("Clear 3rd bit :%x\n",num & ~(1<<3));
    printf("Clear 9th bit :%x\n",num & ~(1<<9));
    printf("Clear 12th bit :%x\n",num & ~(1<<12));
}
```

Assignment2 (Using Left shift operator)

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

```
#include <stdio.h>
int main()
{
    int num;
    printf("Enter number:");
    scanf("%x",&num);
    int n;
    printf("Enter value of n:");
    scanf("%d",&n);
    printf("The product of %d * 2^%d is %d",num,n,num<<n);
}
```

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).

```

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

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

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

    printf("The number of left shifts before overflow: %d\n", 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.

```

#include <stdio.h>
int main()
{
    int num;
    printf("Enter number:");
    scanf("%x",&num);
    int n;
    printf("Enter value of n:");
    scanf("%d",&n);
    int bitmask = (1<<n)-1;
    printf("The bitmask is %d",bitmask);
}

```

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

```

#include <stdio.h>

int main()
{
    unsigned int num, reversed_num;
    printf("Enter a number: ");
    scanf("%d", &num);

    for (int i = 0; i < 32; i++)
    {
        if (num & 1)
        {
            reversed_num |= (1 << (32 - 1 - i));
        }
    }
}

```

```

        num >>= 1;
    }

```

```

    printf("After reversing the number is: %u", reversed_num);
}

```

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

```

#include <stdio.h>

int circular_left(int, int);

int print_bits(int);

int main()
{
    int num, n, ret;

    printf("Enter the num:");

    scanf("%d", &num);

    printf("Enter n:");

    scanf("%d", &n);

    ret = circular_left(num, n);

    printf("Result in Binary :");

    print_bits(ret);
}

int circular_left(int num,int n)
{
    return (((((1<<n)-1) << 31-n) & num) >> (31-n)) | (num<<n));
}

int print_bits(int ret)
{

```

```

for(int i=31;i>=0;i--)

{

    if(ret & 1<<i)

    {

        printf("1 ");

    }

    else

    {

        printf("0 ");

    }

}

}

```

Assignment 3(Using Right shift operator)

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;
    printf("Enter number:");
    scanf("%x",&num);
    int n;
    printf("Enter value of n:");
    scanf("%d",&n);
    printf("The product of %d / 2^%d is %d",num,n,num>>n);
}

```

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()
```

```
{
```

```
    unsigned int num,count=0;
```

```
    printf("Enter the number: ");
```

```
    scanf("%d", &num);
```

```
    int backup_num=num;
```

```
    while(num > 0)
```

```
    {
```

```
        num>>=1;
```

```
        ++count;
```

```
    }
```

```
    printf("%d can be right shifted %d times before turning 0", backup,count);
```

```
    return 0;
```

```
}
```

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

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

```
int main()
```

```
{
```

```
    int num, n, res = 0;
```

```
    printf("Enter a number: ");
```

```
    scanf("%d", &num);
```



```

printf("The number of bits to be extracted: ");

scanf("%d", &n);

for (int i = 0; i < n; i++)

{

    res |= ((num >> i) & 1) << i;

}

for (int i = n - 1; i >= 0; i--)

{

    if (res & (1 << i))

        printf("1");

    else

        printf("0");

}

return 0;

}

```

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.

```

int main()
{
    int num, n;
    printf("Enter the number: ");
    scanf("%d", &num);
    printf("Enter the bit to be checked: ");
    scanf("%d", &n);

    if ((num >> n) & 1)
        printf("%d bits of %d is set\n", n, num);
    else
        printf("%d bits of %d is not set\n", n, num);

}

```