BITWISE

```
srilatha@GESLMP22WP7T:~/Assignment/Bitwise$ cat 1AB.c
#include<stdio.h>

#if QUES == 2
int main()
{
    unsigned char num;
    int i;
    num = 0xAB;
    for (i = 0; i < 8; i++) {
    if (num & 128) {
        printf("1");
    } else {
        printf("0");
    }
    num = num << 1;
    }
    printf("\n");
    num = ((num << 4) & (num >> 4));
    for (i = 0; i < 8; i++) {
        if (num & 128) {
        printf("1");
        } else {
            printf("0");
        }
        num <<= 1;
    }
    printf("\n");
```

```
return 0;
}
#endif


#if QUES == 1
int main()
{
    unsigned char num;
    num = 0xAB;
    int i;
    for (i = 0; i < 4; i++) {
        if (num & 128) {
            printf("1");
        } else {
            printf("0");
        }
      num <<= 1;
    }
    for( i = 0; i < 4; i++) {
        if (num & 3) {
            printf("1");
        } else {
            printf("0");
        }
        num <<= 1;
    }
return 0;
```

```
}
#endif
srilatha@GESLMP22WP7T:~/Assignment/Bitwise$ cat hextobinary.c
#include<stdio.h>

//unsigned int swap_bits_within(unsigned int num, unsigned int s, unsigned int d);
int binary(unsigned int num);

int main()
{
	unsigned int num = 0xABCD;
	/*unsigned int s;
	unsigned int d;
	printf("Enter the number:");
	scanf("%d", &num);
	printf("Enter the first postion to swap:");
	scanf("%d", &s);
	printf("Enter the second position to swap:");
	scanf("%d", &d);*/
	binary(num);
	num = (num >> 12) | (num << 12) ) | ((num >> 4) | (((num << 8) | (num << 4)));
	binary(num);
	//swap_bits_within(num, s, d);
	return 0;
}
int binary(unsigned int num)
{
	int i;
	for (i = 0; i < 16; i++) {
        if (num & 32768) {
            printf("1");
        } else {
            printf("0");
        }
    num <<= 1;
    }
    printf("\n");
}
/*unsigned int swap_bits_within(unsigned int num, unsigned int s, unsigned int d)
{
    if (((num >> s) & 1) && ((num >> d) & 1)) {
        printf("Bits are same");
    } else {
        num = (num ^ ((1 << s) | (1 << d)));
    }

        binary(num);
        printf("\n");
}*/

srilatha@GESLMP22WP7T:~/Assignment/Bitwise$ cat bitwise_display.c
#include<stdio.h>
void bit(char);
int main()
{
    unsigned char num; //= 7;
    printf("Enter the number:");
    scanf("%hhd", &num);
```

```
        bit(num);

        return 0;

}

void bit(char num)

{

        int i;

        for (i = 0; i < 8; i++) {

                if (num & 128) {

                        printf("1");

                } else {

                        printf("0");

                }

        num <<= 1;

        }

}


srilatha@GESLMP22WP7T:~/Assignment/Bitwise$ cd assign/

srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ ls

10setbits.c    1swapbits.c       3nbitstodnum.c  5define.c.save  8countbits.c        copybits   exam
invert  setbitp  swapbetween

11invertbits.c  2swapbitsbetween.c  4toggle.c      6rotate.c      9bitwiseoperations.c  count     exam.c
nbits   setbits  toggle

12getbits.c    3copybits.c       5define.c      7countbits.c    bitwise            countbits  getbits  rotate  swap

srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat 1swapbits.c

#include<stdio.h>


unsigned int swap_bits_within(unsigned int num, unsigned int s, unsigned int d);

int binary(unsigned int num);
int main()
{
        unsigned int num;
        unsigned int s;
        unsigned int d;
        printf("Enter the number:");
        scanf("%d", &num);
        printf("Enter the first postion to swap:");
        scanf("%d", &s);
        printf("Enter the second position to swap:");
        scanf("%d", &d);
        binary(num);
        swap_bits_within(num, s, d);
        return 0;
}
int binary(unsigned int num)
{
        int i;
        for (i = 0; i < 8; i++) {
                if (num & 128) {
                        printf("1");
                } else {
                        printf("0");
                }
        num <<= 1;
        }
        printf("\n");
}
unsigned int swap_bits_within(unsigned int num, unsigned int s, unsigned int d)
```

```
{
        if (((num >> s) & 1) && ((num >> d) & 1)) {
                printf("Bits are same");
        } else {
                num = (num ^ ((1 << s) | (1 << d)));
        }

                binary(num);
                printf("\n");
}

srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat 2swapbitsbetween.c
#include<stdio.h>

int swap_bits_between(unsigned int snum, unsigned int dnum, unsigned int s, unsigned int d);
int binary(unsigned int);
int binary1(unsigned int);

int main()
{
        unsigned int snum;
        unsigned int dnum;
        unsigned int s;
        unsigned int d;
        printf("Enter the source number:");
        scanf("%d", &snum);
        printf("Enter the postion to swap in source:");
        scanf("%d", &s);
        printf("Enter the destination number:");
```

```
    scanf("%d",&dnum);
    printf("Enter the position to swap in destination:");
    scanf("%d", &d);
    printf("Source number is :");
    binary(snum);
    printf("Destination number is :");
    binary1(dnum);
    printf("Number after swapping is :");
    swap_bits_between(snum, dnum, s, d);

    return 0;
}
int binary(unsigned int snum)
{
    int i;
    for (i = 0; i < 8; i++) {
        if (snum & 128) {
            printf("1");
        } else {
            printf("0");
        }
    snum <<= 1;
    }
    printf("\n");
}
int binary1(unsigned int dnum)
{
    int i;
    for (i = 0; i < 8; i++) {
        if (dnum & 128) {
            printf("1");
        } else {
            printf("0");
        }
    dnum <<= 1;
    }
    printf("\n");
}
int swap_bits_between(unsigned int snum, unsigned int dnum, unsigned int s, unsigned int d)
{
    if (((snum >> s) & 1) == ((dnum >> d) & 1)) {
        printf("Bits are same");
        printf("\n");
        return 0;
    } else {
        snum = (snum ^ (1 << s));
        dnum = (dnum ^ (1 << d));
    }

        binary(snum);
        binary1(dnum);
        printf("\n");
}
```

```
srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat 3
cat: 3: No such file or directory
srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat 3
3copybits.c     3nbitstodnum.c
```

```
srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat 3
3copybits.c     3nbitstodnum.c
srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat 3nbitstodnum.c
#include<stdio.h>

unsigned int copy_bits (unsigned int snum, unsigned int dnum,unsigned int n, unsigned int s, unsigned int d);
int binary(int);
int binary1(int);

int main()
{
        unsigned int snum;
        unsigned int dnum;
        unsigned int n;
        unsigned int s;
        unsigned int d;
        printf("Enter source number :");
        scanf("%d",&snum);
        printf("Enter bit position in source :");
        scanf("%d",&s);
        printf("Enter destination number :");
        scanf("%d",&dnum);
        printf("Enter bit position in destination :");
        scanf("%d",&d);
        printf("Enter number of bits:");
        scanf("%d",&n);
        binary(snum);
        binary1(dnum);
        copy_bits (snum, dnum, n, s, d);
return 0;
}

int binary(int snum)
{
        int i;
        for (i = 0; i < 8; i++) {
                if (snum & 128) {
                        printf("1");
                } else {
                        printf("0");
                }
        snum <<= 1;
        }
        printf("\n");
}

int binary1(int dnum)
{
        int i;
        for (i = 0; i < 8; i++) {
                if (dnum & 128) {
                        printf("1");
                } else {
                        printf("0");
                }
        dnum <<= 1;
        }
```

```
    printf("\n");
}
unsigned int copy_bits (unsigned int snum, unsigned int dnum, unsigned int n, unsigned int s, unsigned int d)
{
    int i;
    for(i = 1; i <= n; i++) {
        if(((snum >> (s - n + i)) & 1) !=  (( dnum >> (d - n + i)) & 1)) {
            dnum = (1 << (d - n + i)) ^ dnum;
        }
    }
    //binary(snum);
    binary1(dnum);
return 0;
}
srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat 3copybits.c
#include<stdio.h>
unsigned int copy_bits(unsigned int, unsigned int, unsigned int, unsigned int, unsigned int);
int main()
{
    unsigned int snum;
    unsigned int dnum;
    unsigned int s;
    unsigned int d;
    unsigned int n;
    printf("\nEnter source and destination numbers: \n");
    scanf("%d %d", &snum, &dnum);
    printf("\nEnter the position in source:\n");
    scanf("%d", &s);
```

```
        printf("\nEnter the position in destination:\n");

        scanf("%d", &d);

        printf("\nEnter number of bits to copy:\n");

        scanf("%d", &n);

        unsigned int res = copy_bits(snum, dnum, n, s, d);

        for ( int i = 1 << 7; i > 0; i = i / 2)

        {

                (res & i) ? printf("1") : printf("0");

        }

        return 0;

}

unsigned int copy_bits(unsigned int snum, unsigned int dnum, unsigned int n, unsigned int s, unsigned int d)

{

    unsigned int numbits = sizeof(unsigned int) * 8;

    unsigned int ones_mask;

        unsigned int ones_mask1;

    ones_mask = ((~(unsigned int ) 0) >> (numbits - n)) << s;

        ones_mask1 = ((~(unsigned int ) 0) >> (numbits - n)) << d;

    dnum = (dnum & ~ones_mask1) | (snum & ones_mask);

        return dnum;

}

srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat 4toggle.c

#include<stdio.h>


int binary(unsigned int num);

unsigned int toggle_even_bits(unsigned int num);

unsigned int toggle_odd_bits(unsigned int num);
```

```
int main()
{
    unsigned int num;
    printf("Enter the number:");
    scanf("%d", &num);
    printf("Number is :");
    binary(num);
    printf("After toggle of even bits of given number :");
    toggle_even_bits(num);
    printf("After toggle of odd bits of given number :");
    toggle_odd_bits(num);
    return 0;
}
int binary(unsigned int num)
{
    int i;
    for (i = 0; i < 8; i++) {
        if (num & 128) {
            printf("1");
        } else {
            printf("0");
        }
        num <<= 1;
    }
    printf("\n");
}

unsigned int toggle_even_bits(unsigned int num)
{
    /*int i;
    int count = 0;
    int j = 0;
    for (i = 0; i < 8; i++) {
        if (count % 2 == 0) {
            j = (j ^ (1 << count));
        }
    count = count + 1;
    }
    num = num ^ j;*/
    num = num ^ (1 | (1 << 2) | (1 << 4) | (1 << 6));
    binary(num);
}

unsigned int toggle_odd_bits(unsigned int num)
{
    /*int i;
    int count = 1;
    int j = 0;
    for (i = 0; i < 8; i++) {
        if (count % 2 != 0) {
            j = (j ^ (1 << count));
        }
    count = count + 1;
    }
    num = num ^ j;*/
    num = num ^ ((1 << 1) | (1 << 3) | (1 << 5) | (1 << 7));
    binary(num);
}
```

```
srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat 5define.c
#include<stdio.h>

#define test_set(num,p) if ((num >> p) & 1) {\
                                printf("bit is already set\n");\
                        } else {\
                                num = num ^ (1 << p);\
                                binary(num);\
                        }\

int binary(unsigned int num);

int main()
{
    unsigned int num;
    unsigned int s;
    unsigned int p;
    printf("Enter the number:");
    scanf("%d", &num);
    printf("Enter the position of bit to set :");
    scanf("%d", &p);
    printf("Number is :");
    binary(num);
    printf("setting a bit : ");
    test_set(num, p);
    return 0;
}
int binary(unsigned int num)
{
        int i;
        for (i = 0; i < 8; i++) {
                if (num & 128) {
                        printf("1");
                } else {
                        printf("0");
                }
        num <<= 1;
        }
        printf("\n");
}
srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat 6rotate.c
#include<stdio.h>

unsigned int left_rotate_bits(unsigned int num, unsigned int n);
unsigned int right_rotate_bits(unsigned int num, unsigned int n);
int binary(unsigned int num);

int main()
{
        unsigned int num;
        unsigned int n;
        printf("Enter the number:");
        scanf("%d", &num);
        printf("Enter the number of bits to be rotated:");
        scanf("%d", &n);
        printf("The number is :");
        binary(num);
        printf("After performing left shift on number :");
```

```
    left_rotate_bits(num, n);

    printf("After performing right shift on number :");

    right_rotate_bits(num, n);

    return 0;

}

int binary(unsigned int num)

{

    int i;

    for (i = 0; i < 8; i++) {

        if (num & 128) {

            printf("1");

        } else {

            printf("0");

        }

    num <<= 1;

    }

    printf("\n");

}

unsigned int left_rotate_bits(unsigned int num, unsigned int n)

{

    //num =

    binary((num >> (8 - n)) | (num << n));

    /*if (((num >> s) & 1) && ((num >> d) & 1)) {

        printf("Bits are same");

    } else {

        num = (num ^ ((1 << s) | (1 << d)));

    } */


    //     binary(num);
```

```
                printf("\n");
}

unsigned int right_rotate_bits(unsigned int num, unsigned int n)
{

        //num =
        binary((num >> n) | ((num << (8 - n))));
        //binary(num);
}
srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat 7countbits.c
#include<stdio.h>

unsigned int count_set_bits(unsigned int num);
unsigned int count_clear_bits(unsigned int num);
int binary(unsigned int num);

int main()
{
        unsigned int num;
        printf("Enter the number:");
        scanf("%d", &num);
        binary(num);
        printf("Number of set bits in number are :");
        printf("%d\n", count_set_bits(num));
        printf("Number of clear bits in number are :");
        printf("%d\n", count_clear_bits(num));
        return 0;
}
```

```
int binary(unsigned int num)
{
        int i;
        for (i = 0; i < 8; i++) {
                if (num & 128) {
                        printf("1");
                } else {
                        printf("0");
                }
        num <<= 1;
        }
        printf("\n");
}
unsigned int count_set_bits(unsigned int num)
{
        int i;
        int count = 0;
        /*for (i = 0; i < 8; i++) {
                if (num & 128) {
                        count += 1;
                }// else {
                //}
        num <<= 1;
        }*/
        count = ((num &1)+((num>>1) & 1)) +((num>>2) & 1)+((num >>3) & 1)+((num>>4) &
1)+((num>>5)&1)+(num>>6 &1)+((num>>7 & 1));
return count;
}
```

```
unsigned int count_clear_bits(unsigned int num)
{
        int i;
        int count1 = 0;
        /*for (i = 0; i < 8; i++) {
                if (num & 128) {
                        //printf("1");
                } else {
                //      printf("0");
                        count1 += 1;
                }
        num <<= 1;
        }*/
        i = ((num &1)+((num>>1) & 1)) +((num>>2) & 1)+((num >>3) & 1)+((num>>4) &
1)+((num>>5)&1)+(num>>6 &1)+((num>>7 & 1));
        count1 = 8 - i;
return count1;
}
srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat 8countbits.c
#include<stdio.h>

unsigned int count_leading_set_bits(unsigned int num);
unsigned int count_leading_clear_bits(unsigned int num);
unsigned int count_trailing_set_bits(unsigned int num);
unsigned int count_trailing_clear_bits(unsigned int num);
int binary(unsigned int num);

int main()
{
        unsigned int num;
        unsigned int s;
        unsigned int d;
        printf("Enter the number:");
        scanf("%d", &num);
        binary(num);
        printf("\nNumber of leading set bits are :");
    printf("%d",count_leading_set_bits(num));
        printf("\nNumber of leading clear bits are :");
    printf("%d",count_leading_clear_bits(num));
        printf("\nNumber of trailing set bits are :");
    printf("%d\n",count_trailing_set_bits(num));
        printf("Number of trailing clear bits are :");
    printf("%d\n",count_trailing_clear_bits(num));
        return 0;
}
int binary(unsigned int num)
{
        int i;
        for (i = 0; i < 8; i++) {
                if (num & 128) {
                        printf("1");
                } else {
                        printf("0");
                }
        num <<= 1;
        }
        printf("\n");
}
```

```
unsigned int count_leading_set_bits(unsigned int num)
{
    int i;
    int count = 0;
    for (i = 0; i < 8; i++) {
        if ((num & 128) & (1 << 7)) {
            count++;
        } else {
            return count;
        }
    num <<= 1;
    }
    return count;
}
unsigned int count_leading_clear_bits(unsigned int num)
{
    int i;
    int count = 0;
    for (i = 0; i < 8; i++) {
        if ((num & 128) & (1 << 7)) {
            return count;
        } else {
            count = count + 1;
        }
    num <<= 1;
    }
    return count;
```

```
}
unsigned int count_trailing_set_bits(unsigned int num)
{
    int i;
    int count = 0;
    for (i = 0; i < 8; i++) {
        if ((num & 1) & 1) {
            count = count + 1;
        } else {
            return count;
        }
    num >>= 1;
    }
    return count;

}
unsigned int count_trailing_clear_bits(unsigned int num)
{
    int i;
    int count = 0;
    for (i = 0; i < 8; i++) {
        if ((num & 1) & 1) {
            return count;
        } else {
            count = count + 1;
        }
    num <<= 1;
    }
    return count;
```

```
}
srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat 9bitwiseoperations.c
#include<stdio.h>

unsigned int maximum(unsigned int snum, unsigned int dnum);
int binary(unsigned int num);

int main()
{
        unsigned int snum;
        unsigned int dnum;
        printf("Enter the first number:");
        scanf("%d", &snum);
        printf("Enter the second number:");
        scanf("%d", &dnum);
        binary(snum);
        binary(dnum);
        maximum(snum, dnum);
        return 0;
}
int binary(unsigned int num)
{
        int i;
        for (i = 0; i < 8; i++) {
                if (num & 128) {
                        printf("1");
                } else {
                        printf("0");
                }
        num <<= 1;
        }
        printf("\n");
}
unsigned int maximum(unsigned int snum, unsigned int dnum)
{
        int i;
        for (i = 0; i < 8; i++) {
                if ((snum & 128) > (dnum & 128)) {
                        printf("snum %d is greater than dnum %d", snum, dnum);
                        break;
                }  else if ((snum & 128) < (dnum & 128)) {
                        printf("dnum %d is greater than snum %d", dnum, snum);
                        break;
                } else if ((snum & 128) == (dnum & 128)) {
                        continue;
                }
                snum <<= 1;
                dnum <<= 1;
                }
return 0;
}
srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat 10setbits.c
#include<stdio.h>

unsigned int set_bits(unsigned int dnum, unsigned int p, unsigned int n, unsigned snum);
int binary(unsigned int snum);
int binary1(unsigned int dnum);
int main()

{

        unsigned int snum;

        unsigned int dnum;

        unsigned int p;

        unsigned int n;

        printf("Enter the source number:");

        scanf("%d", &snum);

        printf("Enter the postion of bit in source number:");

        scanf("%d", &p);

        printf("Enter the number of bits:");

        scanf("%d", &n);

        printf("Enter the destination number:");

        scanf("%d", &dnum);

        binary(snum);

        binary1(dnum);

        set_bits(dnum, p, n, snum);

        return 0;

}

int binary(unsigned int snum)

{

        int i;

        for (i = 0; i < 8; i++) {

                if (snum & 128) {

                        printf("1");

                } else {

                        printf("0");

                }
        snum <<= 1;
        }
        printf("\n");
}
int binary1(unsigned int dnum)
{
        int i;
        for (i = 0; i < 8; i++) {
                if (dnum & 128) {
                        printf("1");
                } else {
                        printf("0");
                }
        dnum <<= 1;
        }
        printf("\n");
}
unsigned int set_bits(unsigned int dnum, unsigned int p, unsigned int n, unsigned snum)
{
        snum = (snum << (7 - p)); //>> (8 - p + n));// & (~(0) >> (8 - n))) << (8 - n); //| ((dnum >> p) << p);
        snum = (snum >> (8 - p + n));

        binary(snum);
        //binary1(dnum);
        return 0;
}
srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat 11invertbits.c
#include<stdio.h>
unsigned int invert_bits(unsigned int num, unsigned int p, unsigned int n);
```

```
int main()
{
  unsigned int num;
    unsigned int p;
    unsigned int n;
    printf("\nEnter the number:\n");
    scanf("%d", &num);
    printf("\nBinary representation of num:\n");
    for ( int i = 1 << 7; i > 0; i = i / 2)
    {
        (num & i) ? printf("1") : printf("0");
    }
    printf("\nEnter position:\n " );
    scanf("%d", &p);
    printf("\nEnter number of bits :\n");
    scanf("%d", &n);
    unsigned int res = invert_bits(num, p, n);
    printf("\nAfter inverting the bits are:\n ");
    for (int i = 1 << 7; i > 0; i = i / 2)
    {
        (res & i) ? printf("1") : printf("0");
    }
    printf("\n");
    return 0;



}
unsigned int invert_bits(unsigned int num, unsigned int p, unsigned int n)
```

```
{
    return   num ^ (~(~0 << n) << (p + 1 - n));
}
srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat 12getbits.c
#include<stdio.h>
unsigned int get_bits(unsigned int num, unsigned int p, unsigned int n);
int main()
{
        unsigned int num;
        unsigned int p;
        unsigned int n;
        printf("\nEnter the number:\n");
        scanf("%d", &num);
        printf("\nBinary representation of num:\n");
        for ( int i = 1 << 7; i > 0; i = i / 2)
        {
                (num & i) ? printf("1") : printf("0");
        }
        printf("\nEnter position:\n " );
        scanf("%d", &p);
        printf("\nEnter number of bits :\n");
        scanf("%d", &n);
        unsigned int res = get_bits(num, p, n);
        printf("\nAfter getting the bits are:\n ");
        for (int i = 1 << 7; i > 0; i = i / 2)
        {
                (res & i) ? printf("1") : printf("0");
        }
        printf("\n");
        return 0;




}
unsigned int get_bits(unsigned int num, unsigned int p, unsigned int n)
{
    return   num & ~((~(~0 << n) << (p + 1 - n)));
}



/*#include<stdio.h>


unsigned int get_bits(unsigned int snum, unsigned int p, unsigned int n);
int binary(unsigned int num);


int main()
{
        unsigned int snum;
        unsigned int p;
        unsigned int n;
        printf("Enter the source number:");
        scanf("%d", &snum);
        printf("Enter the postion of bit in source number:");
        scanf("%d", &p);
        printf("Enter the number of bits:");
        scanf("%d", &n);
        binary(snum);
        get_bits(snum, p, n);
        return 0;
}
int binary(unsigned int snum)
{
        int i;
        for (i = 0; i < 8; i++) {
            if (snum & 128) {
                printf("1");
            } else {
                printf("0");
            }
        snum <<= 1;
        }
        printf("\n");
}
/*unsigned int get_bits(unsigned int snum, unsigned int p, unsigned int n)
{
        int count = 0;
        int count1 = 0;
        int i;
        //for (i = 0; i < 8; i++) {
        while (i < 8) {
            if ((i == p) && (n == count1)) {
                count1 = count1 + 1;
                if (snum & 128/count) {
                    printf("1");
                } else {
                    printf("0");
                }
                i++;
                count = count + 2;
        //snum <<= 1;
            } else {
                i++;
            }
        printf("\n");
}
        //     binary(num);
            printf("\n");
}

        snum = (snum>>p);
        for( i = 0; i < n; i++) {
            if (snum & (1<<n)) {
                printf("1");
            } else {
                printf("0");
          }
        }
        printf("\n");
return 0;
}
unsigned int get_bits(unsigned int num, unsigned int p, unsigned int n)
{
  return   num & ~((~(~0 << n) << (p + 1 - n)));
}*/
```

```
srilatha@GESLMP22WP7T:~/Assignment/Bitwise/assign$ cat exam.c
#include<stdio.h>
int main()
{
        int num;
        int i;
        printf("Enter the num :");
        scanf("%d", &num);
        for(i = 0; i < 8; i++) {
                if (num & 128) {
                        printf("1");
                } else {
                        printf("0");
                }
                num = num << 1;
        }
return 0;
}
```

```
srilatha@GESLMP22WP7T:~/Assignment/c$ cd files/
srilatha@GESLMP22WP7T:~/Assignment/c/files$ ls
 al          bitfields.c-  cmd       data    emp     employee.c   exp.txt    filefunctions    func       read
struct    'tdio.h>'
 bitfields.c   bits        cmdarg.c   dynall   empdata   empstruct.c   file2.txt   filefunctions.c   myname.txt
readdata.c   struct.c  'tr[100]'
srilatha@GESLMP22WP7T:~/Assignment/c/files$ cat bitfields.c
#include<stdio.h>
```

```
struct bits {
        short int a;
        int b : 10;
        char c;
        short d : 10;
        short e : 12;

        };

struct bits1 {
        int a : 17;
        short b : 12;
        int c : 7;
        char d : 4;
        short e : 12;
        char g : 4;
        int h : 12 ;
        char f;

        };
int main()
{
        struct bits b;
        struct bits1 b1;
        printf("size of struct is  %d\n",sizeof(b));
        printf("size of struct is  %d\n",sizeof(b1));
        return 0;
}
srilatha@GESLMP22WP7T:~/Assignment/c/files$ cat cmdarg.c
```

```
#include<stdio.h>

int main(int argc, char *argv[])
{
        int i;
        char **p;
        printf("argc = %d \n ",argc);
        for(i = 0; i < argc; i++) {
                printf("argv[%d] = %s \n",i,argv[i]);
        }
        for(p = argv; *p != NULL; p++) {
                //puts(*p);
                printf("%s \n", *p);
        }
        return 0;
}
srilatha@GESLMP22WP7T:~/Assignment/c/files$ cat e
emp         empdata      employee.c   empstruct.c  exp.txt
srilatha@GESLMP22WP7T:~/Assignment/c/files$ cat employee.c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<stdio_ext.h>

struct emp {
        char name[3];
        int empid;
        int age;
        int sal;
        };

int main()
{
        struct emp *empdata;
        FILE *fp;
        int i;
        char str[20];
        printf("size of structure is : %d", sizeof(struct emp));
        empdata = (struct emp*) malloc (sizeof(struct emp));
        fp = fopen("data","wb");
        if (fp == NULL) {
                printf("error");
                exit(1);
        }
        __fpurge(stdin);
        printf("ENTER THE EMPLOYEE DATA \n");
        printf("Enter the employee name :");
        scanf("%s",empdata->name);
        __fpurge(stdin);
        printf("Enter the employee empid :");
        scanf("%d",&empdata->empid);
        __fpurge(stdin);
        printf("Enter the employee age :");
        scanf("%d",&empdata->age);
        __fpurge(stdin);
        printf("Enter the employee salaray :");
        scanf("%d",&empdata->sal);
        //__fpurge(stdin);
        //scanf("%s %d %d %d",empdata->name, &empdata->empid, &empdata->age, &empdata->sal);
        fwrite(empdata,sizeof(struct emp),1,fp);
        printf("EMP NAME  \tEMP ID\tEMP AGE\tEMP SALARY");
        printf("\n");
        printf("%s\t",empdata->name) ;
        printf("%d\t",empdata->empid) ;
        printf("%d\t",empdata->age) ;
        printf("%d",empdata->sal);

fclose(fp);
}
srilatha@GESLMP22WP7T:~/Assignment/c/files$ cat e
emp         empdata     employee.c   empstruct.c  exp.txt
srilatha@GESLMP22WP7T:~/Assignment/c/files$ cat empstruct.c
#include<stdio.h>

struct emp {
    char name[10];
    int empid;
    int age;
    int sal;
    };

int main()
{
    struct emp empdata;
    FILE *fp;
```

```
        int i;
        fp = fopen("data","w+");
        //for (i = 1; i < 3; i++) {
            scanf("%s %d %d %d",empdata.name, &empdata.empid, &empdata.age, &empdata.sal);
        //for (i = 1; i < 3; i++) {
            fwrite(&empdata,sizeof(empdata),1,fp);
        //}
        fclose(fp);
        fp = fopen("data","r");
        while(fread(&empdata,sizeof(empdata) ,1,fp) == 1) {
            printf("%s\t",empdata.name) ;
            printf("%d\t",empdata.empid) ;
            printf("%d\t",empdata.age) ;
            printf("%d\t",empdata.sal);
        }

fclose(fp);
}
```

```
srilatha@GESLMP22WP7T:~/Assignment/c/files$ cat filefunctions.c
#include<stdio.h>
#include <stdlib.h>

#if 1
int main()
{
        FILE *fp;
        fp = fopen("myname.txt","w");


        }
        fclose(fp);
}
#endif

#if 0
int main()
{
        FILE *fp;
        fp = fopen("myname.txt","w");
        while (fscanf(fp) != EOF) {
                printf("%s");
        }
        fclose(fp);
}
#endif

#if 0
int main()
{
        FILE *fp;
        fp = fopen("myname.txt","w");
        fprintf(fp,"my name is srilatha.my name is srilatha.abcdefghijklmnopqrstuvwxyz");
        fclose(fp);
}
#endif
```

```
#if 0
int main()
{
        FILE *fp;
        char c[10];
        fp = fopen("myname.txt","r");
        printf("%s",fgets(c,10,fp));
        fclose(fp);
}
#endif
#if 0
int main() {
        FILE *fp;
        char c[10]; //= "sri";;
        fp = fopen("myname.txt", "w+");
        fgets(c,10,stdin);
        fputs(c, fp);
        fseek(fp , 0, SEEK_SET);
        printf("%s",fgets(c,10,fp));
        fclose(fp);
}
#endif


#if 0
int main()
{
        FILE *fptr1;
        FILE *fptr2;
        char c;
        fptr1 = fopen("myname.txt", "r");
        if (fptr1 == NULL)  {
                printf("Cannot open file \n");
                exit(0);
        }
        fptr2 = fopen("file2.txt", "w");
        if (fptr2 == NULL) {
                printf("Cannot open file\n");
                exit(0);
        }
        c = fgetc(fptr1);
        while (c != EOF) {
                fputc(c,fptr2);
                c = fgetc(fptr1);
        }
        fclose(fptr1);
        fclose(fptr2);
        return 0;
}
#endif



#if 0
int main()
{
        FILE *fp = fopen("myname.txt", "r");
```

```
    int count = 0;
    int word = 0;
    char ch;
    while ((ch = fgetc(fp)) != EOF) {
    if (ch == ' ' || ch == '\n') {
        word = 0;
        } else {
        if (!word){
        count++;
        word = 1;
        }
    }
    }
    printf("%d", count);
    fclose(fp);
    return 0;
}
#endif

#if 0
int main()
{
    FILE *fptr1;
    FILE *fptr2;
    char c;
    fptr1 = fopen("myname.txt", "r");
    if (fptr1 == NULL)
    {
        printf("Cannot open file \n");
        exit(0);
    }
    fptr2 = fopen("file2.txt", "a");
    if (fptr2 == NULL)
    {
        printf("Cannot open file\n");
        exit(0);
    }
    // Read contents from file
    c = fgetc(fptr1);
    while (c != EOF)
    {
        fputc(c,fptr2);
        c = fgetc(fptr1);
    }
    printf("\nContents copied to file2.txt");
    fclose(fptr1);
    fclose(fptr2);
    return 0;
}

#endif

#if 0

int main()
{
    FILE *fp;
    FILE *fp2;
```

```
        char name2[10];
        fp = fopen("myname.txt","r");
        fp2 = fopen("myname2.txt","w");
        name2 = fgets(fp);
        fputs(name2,fp2);
        printf("%c\n",fgetc(fp2));
        fclose(fp);
}
#endif

#if 0
int main()
{
        FILE *fp;
        char name[10];
        fp = fopen("myname.txt","r");
        printf("%s\n",fgets(name,10,fp));
        fclose(fp);
}

#endif

#if 0
int main()
{
        FILE *fd;
        char name[10] = "SIRI latha";
        fd = fopen("myname.txt","w+");
        fputs(name,fd);
```

```
	//printf("%s",fgets(name,5,fd));

	fclose(fd);

	//return 0;

}

#endif

#if 1

int main()

{

	FILE *fp;

	fp = fopen("myname.txt","w");

	while (fscanf(fp) != EOF) {

		printf("%s");

	}

	fclose(fp);

}

#endif


#if 0

int main()

{

	FILE *fp;

	fp = fopen("myname.txt","w");

	fprintf(fp,"my name is srilatha.my name is srilatha.abcdefghijklmnopqrstuvwxyz");

	fclose(fp);

}

#endif
```

```
#if 0
int main()
{
	FILE *fp;
	char c[10];
	fp = fopen("myname.txt","r");
	printf("%s",fgets(c,10,fp));
	fclose(fp);
}
#endif


#if 0
int main() {
	FILE *fp;
	char c[10]; //= "sri";;
	fp = fopen("myname.txt", "w+");
	fgets(c,10,stdin);
	fputs(c, fp);
	fseek(fp , 0, SEEK_SET);
	printf("%s",fgets(c,10,fp));
	fclose(fp);
}
#endif


#if 0
int main()
{
    FILE *fptr1;

    FILE *fptr2;

    char c;

    fptr1 = fopen("myname.txt", "r");

    if (fptr1 == NULL)  {

        printf("Cannot open file \n");

        exit(0);

    }

    fptr2 = fopen("file2.txt", "w");

    if (fptr2 == NULL) {

        printf("Cannot open file\n");

        exit(0);

    }

    c = fgetc(fptr1);

    while (c != EOF) {

        fputc(c,fptr2);

        c = fgetc(fptr1);

    }

    fclose(fptr1);

    fclose(fptr2);

    return 0;

}

#endif




#if 0

int main()

{

    FILE *fp = fopen("myname.txt", "r");
```

```
        int count = 0;

        int word = 0;

        char ch;

        while ((ch = fgetc(fp)) != EOF) {

        if (ch == ' ' || ch == '\n') {

                word = 0;

                } else {

                if (!word){

                count++;

                word = 1;

                }

        }

        }

        printf("%d", count);

        fclose(fp);

        return 0;

}

#endif


#if 0

int main()

{

        FILE *fptr1;

        FILE *fptr2;

        char c;

        fptr1 = fopen("myname.txt", "r");

        if (fptr1 == NULL)

        {

                printf("Cannot open file \n");
                exit(0);
        }
        fptr2 = fopen("file2.txt", "a");
        if (fptr2 == NULL)
        {
                printf("Cannot open file\n");
                exit(0);
        }
        // Read contents from file
        c = fgetc(fptr1);
        while (c != EOF)
        {
                fputc(c,fptr2);
                c = fgetc(fptr1);
        }
        printf("\nContents copied to file2.txt");
        fclose(fptr1);
        fclose(fptr2);
        return 0;
}

#endif

#if 0

int main()
{
        FILE *fp;
        FILE *fp2;
```

```
        char name2[10];

        fp = fopen("myname.txt","r");

        fp2 = fopen("myname2.txt","w");

        name2 = fgets(fp);

        fputs(name2,fp2);

        printf("%c\n",fgetc(fp2));

        fclose(fp);

}

#endif


#if 0

int main()

{

        FILE *fp;

        char name[10];

        fp = fopen("myname.txt","r");

        printf("%s\n",fgets(name,10,fp));

        fclose(fp);

}


#endif


#if 0

int main()

{

        FILE *fd;

        char name[10] = "SIRI latha";

        fd = fopen("myname.txt","w+");

        fputs(name,fd);
        //printf("%s",fgets(name,5,fd));
        fclose(fd);
        //return 0;
}
#endif
#if 1
int main()
{
        FILE *fp;
        fp = fopen("myname.txt","w");
        while (fscanf(fp) != EOF) {
                printf("%s");
        }
        fclose(fp);
}
#endif

#if 0
int main()
{
        FILE *fp;
        fp = fopen("myname.txt","w");
        fprintf(fp,"my name is srilatha.my name is srilatha.abcdefghijklmnopqrstuvwxyz");
        fclose(fp);
}
#endif
```

```
#if 0
int main()
{
	FILE *fp;
	char c[10];
	fp = fopen("myname.txt","r");
	printf("%s",fgets(c,10,fp));
	fclose(fp);
}
#endif
#if 0
int main() {
	FILE *fp;
	char c[10]; //= "sri";;
	fp = fopen("myname.txt", "w+");
	fgets(c,10,stdin);
	fputs(c, fp);
	fseek(fp , 0, SEEK_SET);
	printf("%s",fgets(c,10,fp));
	fclose(fp);
}
#endif


#if 0
int main()
{
	FILE *fptr1;
	FILE *fptr2;
```

```
    char c;
    fptr1 = fopen("myname.txt", "r");
    if (fptr1 == NULL)  {
        printf("Cannot open file \n");
        exit(0);
    }
    fptr2 = fopen("file2.txt", "w");
    if (fptr2 == NULL) {
        printf("Cannot open file\n");
        exit(0);
    }
    c = fgetc(fptr1);
    while (c != EOF) {
        fputc(c,fptr2);
        c = fgetc(fptr1);
    }
    fclose(fptr1);
    fclose(fptr2);
    return 0;
}
#endif



#if 0
int main()
{
    FILE *fp = fopen("myname.txt", "r");
    int count = 0;
    int word = 0;
```

```
        char ch;
        while ((ch = fgetc(fp)) != EOF) {
        if (ch == ' ' || ch == '\n') {
                word = 0;
                } else {
                if (!word){
                count++;
                word = 1;
                }
        }
        }
        printf("%d", count);
        fclose(fp);
        return 0;
}
#endif

#if 0
int main()
{
        FILE *fptr1;
        FILE *fptr2;
        char c;
        fptr1 = fopen("myname.txt", "r");
        if (fptr1 == NULL)
        {
                printf("Cannot open file \n");
                exit(0);
        }
```

```
    fptr2 = fopen("file2.txt", "a");
    if (fptr2 == NULL)
    {
        printf("Cannot open file\n");
        exit(0);
    }
    // Read contents from file
    c = fgetc(fptr1);
    while (c != EOF)
    {
        fputc(c,fptr2);
        c = fgetc(fptr1);
    }
    printf("\nContents copied to file2.txt");
    fclose(fptr1);
    fclose(fptr2);
    return 0;
}

#endif

#if 0

int main()
{
    FILE *fp;
    FILE *fp2;
    char name2[10];
    fp = fopen("myname.txt","r");
```

```
        fp2 = fopen("myname2.txt","w");

        name2 = fgets(fp);

        fputs(name2,fp2);

        printf("%c\n",fgetc(fp2));

        fclose(fp);

}

#endif


#if 0

int main()

{

        FILE *fp;

        char name[10];

        fp = fopen("myname.txt","r");

        printf("%s\n",fgets(name,10,fp));

        fclose(fp);

}


#endif


#if 0

int main()

{

        FILE *fd;

        char name[10] = "SIRI latha";

        fd = fopen("myname.txt","w+");

        fputs(name,fd);

        //printf("%s",fgets(name,5,fd));

        fclose(fd);
```

```
    //return 0;
}
#endif
srilatha@GESLMP22WP7T:~/Assignment/c/files$ cat readdata.c
#include<stdio.h>
#include<stdlib.h>


struct emp {
    char name[10];
    int empid;
    int age;
    int sal;
    };
int main()
{
    struct emp empdata;
    FILE *fp;
    char str[100];
    fp = fopen("data","rb");
    while(fread(&empdata,sizeof(empdata) ,1,fp) == 1) {
        printf("%s\t",empdata.name) ;
        printf("%d\t",empdata.empid) ;
        printf("%d\t",empdata.age) ;
        printf("%d\t",empdata.sal);
    }

fclose(fp);
}
```

```
srilatha@GESLMP22WP7T:~/Assignment/c/files$ cat struct.c
#include<stdio.h>
//#pragma pack(1)

struct emp {
        char name;
        short t;
        int f;
//      double p;
//      double l;
        long double a;
//      float fq;
//      char j;
        }ep;
int main() {
        printf("size of struct is: %ld\n",sizeof(struct emp));
return 0;
}
```