```
      // STRUCTURE  WITHIN  STRUCTURE


  #include<stdio.h>

      struct    company
      {
              char     name[10];
              int      code;

              struct   branch
              {
                          char bname[10];
      int  bcode;
              }b;
      }c;

int  main()
{

            printf("ENTER COMPANY NAME AND CODE\n");

             scanf("%s%d",c.name,&c.code);
```

```
              printf("ENTER BRANCH NAME AND CODE\n");

            scanf("%s%d",c.b.bname,&c.b.bcode);
printf("company name = %s\n",c.name);

            printf("company code =%d\n",c.code);

            printf("branch name = %s\n",c.b.bname);

            printf("branch code =%d\n",c.b.bcode);
        }
    /*
    ------------------------------------------------------------------
```

union

(COLLECTION OF DIFFERENT DATA TYPE)

with the help of union we can create user-defined data type.

```
            e.g.

            union  boolean // tagname
            {
                    int  true;          // 2 byte

                    int  false;
            } a ;

            size of boolean = 2 byte

  -------------------------------------------------------------------
*/
  //   DIFFERENCE BETWEEN STRUCTURE AND
UNION

#include<stdio.h>
    struct    test1
  {
        int   true ;          //   true   false

        int   false;        //      2      2

  }a;                  //      -- 4 bytes --
```

```
   union     test2
   {
         int   true;

         int   false;       //      2 byte
   }b;

   int   main()
   { printf(" SIZEOF STRUCTURE = %d\n" , sizeof(struct test1));
// 4

 printf(" SIZEOF UNION     = %d\n" , sizeof(union  test2));   //  2

   }
```

USER - DEFINED DATA TYPE

----

enum :- ENUMERATED DATA TYPE

syntax

```
enum   data_type { member1, member2,....... };
```

e.g.

```
1.     enum  boolean { false  , true };
                                                0      1
                  boolean  t;

                  t = false ;  // t = 0

                  t = true ;  //  t = 1


2.     enum  day { sun , mon , tue , wed ,thru , fri , sat };
                                    0      1    2     3     4      5    6
                  day   t;

                  t = mon ;   // t = 1


3.     enum  day { sun , mon , tue = 25 , wed , thru, fri, sat };
                                    0     1        25      26    27
                         day  t;
                          t = tue;  // t = 25
       ---------------------------------------------------------------
```

typedef :-

```
                    //    TO RENAME DATA TYPE

                syntax

                typedef  old-data-type    new-data-type

  e.g
1.   typedef  int  complex ;

               complex  a , b ;  // int  a , b;


2.   typedef  struct  student
               {
                       char name[10];

                       int  roll;
               };

          student  a , b , c ;  // struct  student  a,b,c;   ---------
---------------------------------------------------------------
```

```
//   INPUT  RECORDS AND SORT  N  RECORDS

#include<stdio.h>

  struct    student
  {
        char  name[10];

        int   roll;

  } a[10],t;

  int    main()
  {

             int  i , j,n ;

             printf(" ENTER NO. OF RECORDS \n ");

             scanf("%d", &n);

             for( i = 0 ; i < n ; i++ )
             {
```

```
      printf(" ENTER NAME AND ROLL \n ");
           scanf("%s%d", a[i].name , &a[i].roll);

 } // INPUT  N RECORDS


 for( i = 0 ; i < n ; i++ )
 {
for( j = 0 ; j < n-i-1 ; j++ )
 {
   if(a[j].roll > a[j+1].roll)
   // if(strcmp(a[j].name ,a[j+1].name) > 0 )
   {

      t = a[j];    // SWAP REOCORDS
      a[j]=a[j+1]
      a[j+1]= t;
   }
 }
 }

 printf(" name \t\t roll\n");
 for( i = 0 ; i < n ; i++ )
 {
```

```
                    printf("%s\t\t%d\n" , a[i].name,a[i].roll);

            }
 }
-----------------------------------------------------------------------
            //    INPUT  RECORDS AND SEARCH

PERTICULAR RECORDS

#include<stdio.h>

  struct    student
  {
        char  name[10];

        int   roll;

  } a[10];

  int    main()
  {

             int  i , j , n , x , t;
```

```
          printf(" ENTER NO. OF RECORDS \n ");

          scanf("%d", &n);
          for( i = 0 ; i < n ; i++ )
          {
               printf(" ENTER NAME AND ROLL \n ");

               scanf("%s%d", a[i].name , &a[i].roll);

          } // INPUT  N RECORDS

          printf("enter roll to be searhed\n");
scanf("%d",&x);

          for( i = 0 ; i < n ; i++ )
          {
         if(a[i].roll == x )
           // if(strcmp(a[i].name ,x) == 0 )
          {
            printf(" NAME = %s \n" , a[i].name);

            printf(" ROLL = %d \n" , a[i].roll);

            t = 1 ; break;
            }
             }

    }
```

1. int i = 32770;

```
printf(" %d\n",  i);
```

integer range

-32768 to + 32767

```
 i :- 1, 2, 3, ..... + 32767, -32768 , -32767, -32766, ...., -2, -1,
0 ,1,2,..
                           or      or      or
                          +32768   +32769  +32766
```

ans :- -32766

2.

```
int   a = 400 * 400 / 400 ;

  if (  a == 400 )
  {         printf(" A");
  #include<conio.h>
  }
else
{
       printf(" B ");
  }

  ans :-  B
```

3.

```
int  i;
for ( i = 1 ;  i <= 32767 ; i++ )
{
     printf(" %d \n", i);
}
```

infinite loop

i :- 1, 2, 3, ... + 32767, -32768 , -32767, -32766, ..., -2, 1, 0 ,1,2,..

#include<conio.h>

```
4.   int  i;   for ( i = 1 ;  i

     < 32767 ; i++ )

       {
           printf(" %d \n", i);
       }
```

i :- 1, 2, 3, ... + 32766

```
5.   int  i = 1;
      for ( ; i++ ;  )
      {
           printf( "%d\n", i);
      }
```

i :- 1, 2, 3, ... + 32767, -32768 , -32767, -32766, ..., -2, -1
next i --> 0 false

```
3.  int  i = 32770 ;
    printf(" %d\n" , i); // -32766
```

i :- 1, 2, 3, ... + 32767, -32768 , -32767, -32766, ..., -2, 1, 0 ,1,2,..

```
                          or      or       or
                        +32768    +32769   +32770
```

```
4.  int i = 400 * 400 / 400 ;

    if( i == 400 )
    {
       printf( " A");
    }
    else
    {
      printf(" B");
    }

    ans :-  B
```