Assignment-5 (structures & unions): -

- Create a structure for student/employee information with suitable members and do the following
  - Creating variables, input, output operations
  - initialization(complete or partial) of variables
  - create a pointer of struct type, and assign address of variable
  - access members from the pointer using arrow operator
  - calculate size of variable, offset of each member
  - create alias for the structure type, pointer type using typedef
- For the following structures calculate overall size, offset of each member with different packing options (default, 1, 2 bytes etc.).Estimate the padding bits in each case.

```
struct A {          struct B            struct C
   int x;           {                   {
   short int y;         int x;              int x;
   double d;            char str[5];        short int y;
   float f;             double y;           float y;
  char carr[5];         short int z;        short arr[5];
};                  };                  };
```

  Try out some more structures with character variables, array of characters, short variables,array of short variables with odd no.of elements etc.

- Given the address of a member variable in a structure find the base address of it.

  For eg:- struct A {

```
               int x;
               double y;
               float z;
               char ch;
          }a1;
```

  given address of any member x,y,z or ch , find address(base) of a1.

- Create a Box structure with the members length,breadth,height. Pass the structure variable to a function to calculate surface area by value, by reference. Which is efficient even when modifications are not expected in calling function.

- In the student structure created above modify marks member as an array(array of 5 subjects), create an array of struct variables and do some input,output operations. (Marks of ith student in jth subject etc).Find the student wise totals, subject wise totals.
- Create an employee structure with the following members

  empid, employee name, salary, year of joining etc.

  Accept thedata for certain no.o of employees and find their total, average, max, min salary. Also find out the employees with maximum, minimum service,use a function to find service for each employee element in the array.
- Whats wrong in the following code, suggest a fix for this.

```
strcut A
{
        int x;
        char *str;  // (or) char str[20];
};
 struct A a1 = { 101, “abc” } , a2;
 a1.x=10;
 a1.str=”hello”; //works?
 scanf(“%d%s”,&a1.x,a1.str); //works?
 a2 = a1;  //shallow copy or deep copy?
```

  What if str is declared as an array instead of pointer, i.e. char str[20];
- Can a function return structure variable by value?

  Any better alternatives to this if it is possible or not.

  For eg:- struct box create_cube(int s)

```
{
      struct box b1={s,s,s};
      return b1;
 };
```

Provide a better code to avoid returning structure types by value.

- Create an anonymous structure, create some variables from this (with & without typedef)
- Create a nested structure, access members of inner structure from outer one.
- Create a structure with bit fields,analyze size of structure and range of each member.

  Can you find address or offset for bit field members?
- Use bitfield members to set, reset, flip,query the particular bits in a given number.

Unions:-

- Try the following code

```
 union A
{
   int x;
   int y;
   char ch;
  };
  union A a1;
  a1.x=0x10;      a1.y=0x1121;      print a1.x, a1.ch
```

- Calculate size of union , offset of each member
- Convert ip address between dotted decimal format, 32 bit format using unions
- Analyze the following code

```
union A
{
      int x;
      float y;
      double z;
      int arr[2];
 }a1;
     a1.y=6.25f;
     printf(“x=%x\n”,a1.x);
     a1.z=0.15625;;
```

```
        printf(“%x%x\n”,a1.arr[1],a1.arr[0]);
```

- union B

```
    {
        int x;
        short int y;
        char ch;
        char carr[4];
    }b1;
    b1.x=0x41424344;
```

  Analyze the values of b1.y, b1.ch, print all elements of b1.carr

  Modify b1.y or b1.ch and check the value of b1.x

- Try out anonymous unions and usage of typedef
- Try out nesting of structures, unions

  Union inside a structure

  Structure/Structures inside an union

  Union inside another union etc.