2. Structure & Union

Q.1 Structure: -

Structure is a collection of one more or more time & variable different datatypes group together under a single name for convenient handing.

Ex.

```
Structure student – record
 {
    char name[50];
    int weight;
    float height;
 };
```

- Structure help to organized complex data in the meaning full way.
- Structure permit a group of related variable to be used as separate entities.
- General format of structure.
- Structure must be define first for their format that may be use letter to declare structure variable.

Syntax:-

```
struct <struct name>
{
  data type member name 1;
  data type member name 2;
  data type member name 3;
  ……..;
  ……;
```

};

Ex.

```
struct student record
{
  char name [50];
  int weight age;
  float heigh;
 };
```

- The structure is reminded with semicolon;
- The entire definition is consider as a statement each member is declare independently for it's name and types in a separate statement inside the structure.

Q.2. Declaration of structure variable: -

Ans.

We can declared variable of that type a structure declaration of variable any other datatype.

Syntax:-

```
struct   student record stu1, stu2,stu3;
main()
{
clrscr();
 printf("enter the stdent record= ");
 scanf("%s%d%f",stu1.name,&stu1.weight,&stu1.height);
 scanf("%s%d%f",stu2.name,&stu2.weight,&stu2.height);
```

```
scanf(“%s%d%f”,stu3.name,&stu3.weight,&stu3.height);

printf(“information of student”);

printf(“%s%d%f\n,stu1.name,stu1.weight,stu1.height);
```

printf(“%s%d%f\n,stu2.name,stu2.weight,stu2.height);

printf(“%s%d%f\n,stu3.name,stu3.weight,stu3.height);

getch();

}

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

struct Student
{
    char name[25];
    int age;
    char branch[10];
    char gender;
};

int main()
{
    struct Student s1;

    s1.age = 18;
    strcpy(s1.name, "Viraaj");
    printf("Name of Student 1: %s\n", s1.name);
    printf("Age of Student 1: %d\n", s1.age);

    return 0;
}
```

Q.3. Structure within structure//or// Nesting structure: -

Ans.

In ‘c’ structure within structure is also call nesting structre in program.

- There are two type structure

1. inner structure.

2.outer structure.

Syntax:-

```
struct outer- structure
{
 ……
 ……
struct inner- structure
 {
    Data type inner variable;
    ……
    ……..
 } inner- member inner member-2
}outer member-1,outer mrmber-2;
```

Q.4 Array within structure: -

Ans.

'C' allow of structure in a program we declare on array of structure each member or elements of array representing a structure variable.

Ex.

```
struct marks
{
    int sub1;
    int sub2;
}
 struct  marks student [2] {40,45}, {50,60}
printf("sturct marks = ");
```

```
struct Employee
{
    char ename[10];
    int sal;
};

struct Employee emp[5];
int i, j;
void ask()
{
    for(i = 0; i < 3; i++)
    {
        printf("\nEnter %dst Employee record:\n", i+1);
        printf("\nEmployee name:\t");
        scanf("%s", emp[i].ename);
        printf("\nEnter Salary:\t");
        scanf("%d", &emp[i].sal);
    }
    printf("\nDisplaying Employee record:\n");
    for(i = 0; i < 3; i++)
```

```
    {
        printf("\nEmployee name is %s", emp[i].ename);
        printf("\nSlary is %d", emp[i].sal);
    }
}
void main()
{
    ask();
}
```

Q.5. initialization of structure: -

Ans.

In structure you can also initialization of structure.

- Array of structure મ array main variable string મ declare થ ય.
- Array within structure મ array ની અંદર variable declare થ ય.

Ex.

```
main()
{
    struct personal info(information)
    {
        char name[50]
        int weight;
        float height;
    };
    struct personal info pre1:{“Ram”,45.35};
```



```
struct personal info per2:{“Jay”,37.50}
printf(“%.....);
printf(……..);
```

Or

```
main()
{
    struct personal-info
    {
        char name [50];
        int weight; float
        height;
        per1={“Ram”,40,5.8}
    };
```

```
    struct personal info per={“Jay”,45.5.3};
    printf(“          “);
getch();
}
```

Q.6. Array of structure

Ans.

Ex.

```
struct  student [120]
{
  char name[50];
```

```
int age;
}
```

- We use structure to describe the format of related variable.
- In the structure each element of array representing a structure variable that consist of 120 elements.

Ex.

```
{
   struct   class student[100];
   struct marks;
    {
```

```
    int sub1;
    int sub2;
  };
 struct marks student [2]={140,50},{50,60};
}
```

Q.7. copy & comparing structure variable:-

Ans.

- Two variable of the same structure type can be copy the same way as ordinary variable.

Ex.

```
St2=St1 ;
 if (st2 !=st1)
 if (st2==st1)
```

Ex. struct class

```
  {
 int no;
 char name[20];
 float per;
};
main()
{
 int x;
```

```
    struct class st1={1,"Ram",72.50};
    struct class st2={ 2."Raj",50.5};
    struct class st3;
     st3=st1;                          // copy
      x=(st3.name==st2.name"?:0;

    if (x=1)
   {
      printf ("st1 & st3's name is same");
      printf (%d%s%f",st3.no,st3.name,st3.per);
   }
   else
   {
     Printf("st2 & st3 are not same")
   }
}
```

'C' does not permit any logical operation on structure variable.

Q.8. Union: -

Ans.

Union is same concept like structure there is major difference between structure and union in term of storage.

- In structure at each member has it's own storage location the members of union use the same location.
- A union may contain many members of different types union can only one member at a time.

Syntax: -

```
union (union name)
{
   data type member1;
   data type member2;
    ……..
    ……..
};
```

Ex.: -

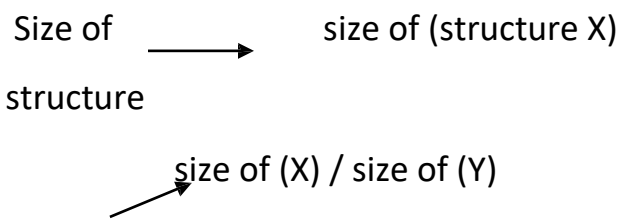
```
}
union
```

- Union created stored location that can be use any one of it's member at a time.

Q.9. Size of structure:-

Ans. we normally use structure union & array to create variable of large size.

```
struct student
{
  int no;
  char name[20];
```



It's main use the un array operator size of structure or variable.

- 8 bit = 1 byte.
- 1024 byte= 1 KB.
- 1024 KB= 1 MB.
- 1025 MB= 1GB.
- 1024 GB= 1TB

Q.10. Bit filed.:-

Ans.

We have been using integer filled of 16 bit data to store in memory.

When data- item required much less then 10 bit space in such case we west memory space. That's why 'C' permit small bits fills to hold data item to allow direct manipulation of string or size of undefined structure.

Syntax:-

```
struct tag name
{
  datatype name: bit length;
  datatype name: bit length;
   ……..
   ……
}
```

Ex.

```
struct personal
 {
   Char name [20];
   Unsigned sex : 1;
   Unsigned age: 5;
 }emp [100];
```

The declare emp is a hundred(100) element array of type struct personal.

Q.11. Structure us union.

Or

Ex. of structure & union memory location.

Ans.

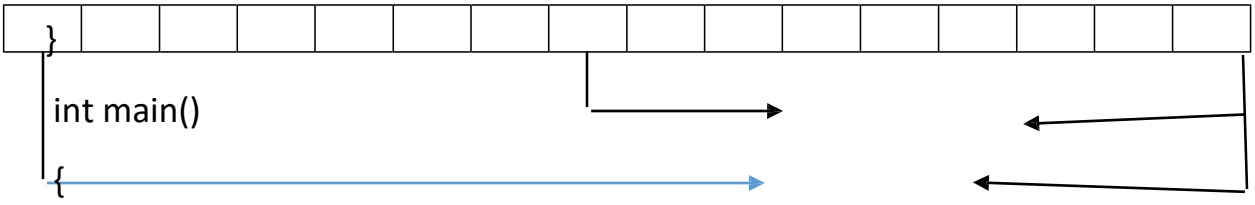
Memory location the amount of memory regularly to store a structure

is the sum of the size of all the member in addition to the bytes that may be provide by the complier.

Ex.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
struct  s
 {
     int I;
     char ch;

   double d;
 };
Union U
 {
   int I;
   char ch;
   double d;
 }
int main()
 {
     printf (“size of structure is%d”,size of (struct s));
     printf (“Size of union is %d”,size of (union u ));
      return(0);
```



}

Q.12. Bit position of union.

Ans.

Bit position

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Chartered

Integer

Q.13. Structure & function.

Ans.

- We know that c language is of function.
- C support the passing of structure value as arguments to function.
- The value of structure can be transfer for one function to another function.

Syntax:-

Data type function_ name (struct- type, st_name)

{

……

……..

Return(expression)

}

- Function must be declare it's type appropriate to the data type it is accepted to return.
- It must be declare as struct with and appropriate tag name.
- The return statement is necessary only when the function is returning some data back to the calling.

Ex.

```
struct student
{
```

```
        char name [50];
         int no;
      };
      int   main()
    {

     struct student s1;
     printf("enter student name= ");
     scanf("%s",s1.name);
     printf("enter no ");
     scanf(%d",&s1.no);
   display (s1);                          // call function
   return 0 ;
 }
 void display (struct student stu)
 {
        printf("name%s",stu.name);
        printf("no%d",std no);
}
```

Q.14. Accessing stricter elements of giving a value to member: -

Ans.

1. Members or external are not a variable.
2. The link between a members or elements and variable is established using members operator.

"." Which is also known as Dot operator or period operator.

Syntax: -

scanf("%d".&stu1.weghit);

Ex.

```
struct student-record
{
    char name[50];
```

```
            int weghit;
            float height;
        };
  struct student record stu1,stu2,stu3;
main()
{
    clrscr();
    printf("enter the student record= ");
    scanf("%s%d%f",stu1.name,&stu1.weghit ,&stu1.height);
    scanf("%s%d%f",stu2.name,&stu2.weghit ,&stu2.height);
    scanf("%s%d%f",stu3.name,&stu3.weight,&stu3.height);
    printf("Information of student");
    printf(%s%D%f",st1.name,st1.weight,st1.height);
    printf(%s%D%f",st2.name,st2.weight,st2.height);
    printf(%s%D%f",st3.name,st3.weight,st3.height);
   getch();
}
```