Assignment -10

09/09/2023

Q] What is the concept of Structure in C programming? Explain with example.

Ans.] Structure is an user-defined data type

] Structure holds heterogenous type data based on programmer's requirement

] Programmer is the user of language

] Structure is an user-defined data type which can store] primitive data type in it

] derived data type in it

] user-detine type in it

] The point where we create structure's layout is called as declaration of structure.

```
struct Demo
{
    int i;
    float j;
};
```

← members of structure

~~char k;~~
~~double d;~~

] When we declare the structure, the compiler will predict the no. of bytes required for that structure but at that time, there is no memory allocation

] Memory for structure gets allocated when we create the objects for structure.

] Syntax : struct Demo obj1;

obj 1 is considered object of structure,

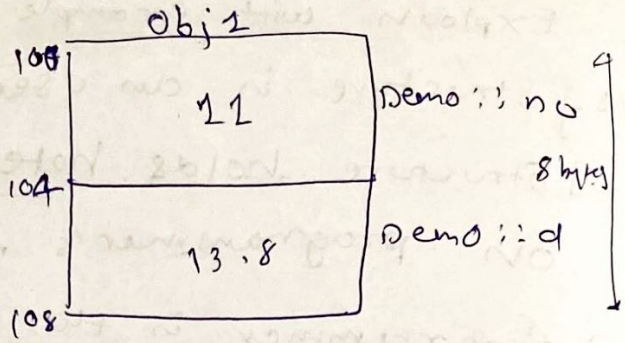
] Size of object is equal to summation of sizes of individual members of structure. (padding may get added).

] To access the members of structure we use '.' (dot) operator also called as direct accessing operator

~~] Initia~~

] Example:

```
~~int main~~
Struct ~~main~~ Demo
{
    int no;
    float d;
};
int main ()
{
    Struct demo obj1;
    obj1.no = 11;
    obj1.d = 13.8;
}
```



Q2] What is the difference between structure & union?

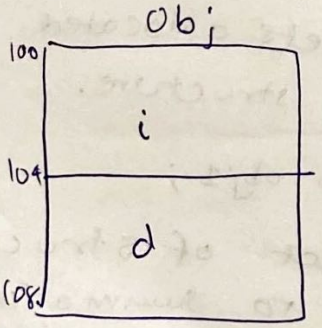
Ans The difference between structure & union is only of memory allocation.

] In case of structure memory gets allocated for each & every member seperately.

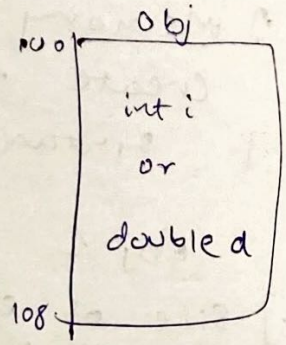
] In case of Union memory gets allocate only for the largest member of union.

] In union we can store only one value at a time.
In structure ~~we can store~~ memory is allocated for value of every member

```
Struct Demo
{
    int i;
    float d;
};
Struct Demo obj;
```




```
Union Demo
{
    int i;
    double d;
};
Union demo obj;
```



Q2] Which data types can we store in structure?

Ans Structure is an user-defined data type.
It can ~~can~~ store derived data types
Primitive data types
& User-Defined data types.

~~Q3] We can initialise.~~

Q3] In how many ways can we initialise members of structure?

Ans
```
struct Demo
{
    int i;
    float d;
} obj1, obj2;
```

```
struct Demo
{ int i;
  float d;
};
int main()
{
  struct Demo obj1;
}
```

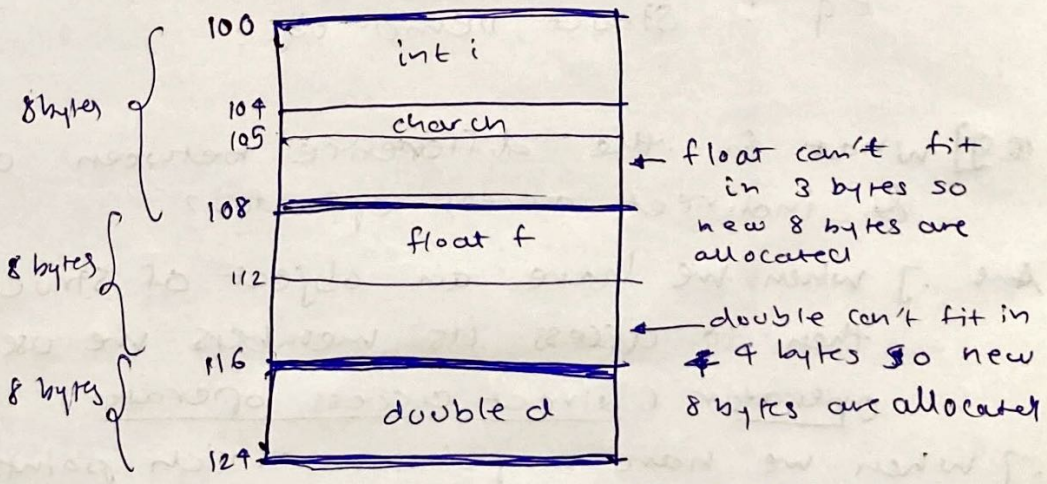
Q4] What is meant by padding in memory allocation of structure?

Ans.) Padding is considered as allocated memory inside the object that we cannot access. It is a wastage of memory.

] To access the elements in a faster way the compiler will assign ~~memory in the fastest way~~ extra memory. This extra memory is called as padding.

Example

```
struct Demo
{
    int i;
    char ch;
    float f;
    double d;
};
```



] In Padding as the largest element is of 8 bytes, memory gets allocated in terms of 8 bytes

] Padding can be avoid by using #pragma pack.

Q6] What is Difference between array & structure?

Ans

Array	Structure
Derived Data type	User Defined Data types
Stores homogenous data only	Stores heterogenous as well as homogenous data type.
Accessed using index arr[5]	Accessed using '.' (dot) direct accessing operator obj1.no.

Q8] Why we cannot initialise members of a structure at the time of declaration?

Ans.] Structure Declaration is the layout of structure.

.] The compiler just predicts the no. of bytes needed for the members of structure but will not allocate memory for it.

.] Since memory is not allocated initialization of members inside structure cannot be done.

.] Memory to structure members is allocated when an object of structure is created.

Eg: Struct Demo obj;

Q9] What is the difference between direct access & indirect access operator?

Ans .] When we have an object of structure or union, then to access its members we use the '.' dot operator (direct access operator)

.] When we have a pointer which points to object of structure / union, then to access its members we use the indirect access operator (→).

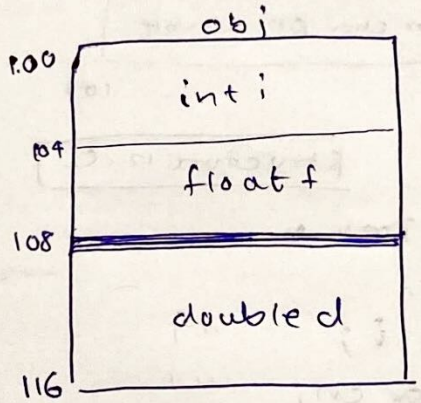
Q10] Detect problem in below syntax

1.
```
struct demo
{
    int i;
    float f = 10.0;     // Initialisation of members
    double d;           //  is not allowed during
};                      //  declaration of structure.
```

2.
```
struct demo
{
    int i;
    float f;
    double i;  // Each member of structure should
               //  have a unique name.
};
```

Q7] Draw memory layout of below structure.

1]
```
struct demo
{
    int i;
    float f;
    double d;
};
```



2]
```
struct demo
{
    int arr[3];
    float f;
    double d;
};
```