SECTION - A
Very Short Answer Questions
3 Marks (50 words each)

Q1. What is Structure?

Ans. Structure is a collection of heterogeneous type of data i.e. different types of data. The various individual data components in a structure can be accessed and processed separately. A structure is a collection of variables referenced under one name, providing a convenient means of keeping related information together. A structure declaration forms a template that may be used to create structure objects (that is, instances of a structure).

Q2. What are the uses of structure?

Ans. The immediate application that comes to the mind is database management. That is, to maintain data about employees in an organization, books in a library, items in a store, financial accounting transaction in a company etc. but mind you, use of structure stretches much beyond database management. They can be used for a variety of applications like-

(a) changing the size of the cursor
(b) clearing the contents of the screen
(c) placing the cursor at appropriate position on screen
(d) drawing any graphics shape on the screen
(e) receiving a key from the keyboard
(f) checking the memory size of the computer
(g) finding out the list of equipments attached to the computer
(h) formatting a floppy
(i) hiding a file from the directory
(j) displaying the directory of a disk
(k) sending the output to printer
(l) interacting with the mouse

Q3. Give the declaration of Structures?

Ans. A structure definition contains a keyword **struct** along with structure name which is followed by the curly braces, which contains different components/ elements/ member of the structure of various data types. The general format of a simple structure declaration is:

```
<storage_class>struct<user_defined_name>
{     datatype      member1;
      datatype      member2;
      .....
      .....
};
```

In above declaration,

Storage_class refers to the scope of structure variable. The storage class is optional **struct** is a keyword which shows the start of a structure.

user_defined_name is the structure name defined by the user.

Q4. How structure variables are declared?

Ans. The general format of declaring structure variables inside the structure is:

```
<storage_class>struct<user_defined_name>
{
      datatype member_1;
      datatype member_n;
}
variable_1, variable_2
```

For example, the object of the structure employee can be declared as:

```
struct employee
{     char name[30];
      char addr[40;]
      int e_no;
      int e_age
} emp_info, binfo, cinfo,
or
struct employee emp_info,binfo,cinfo;
```

Q5. How can we access the fields of an structure?

Ans. The members of the structure themselves are not variables. The members have to be linked with the structure variable so as to make the members

ing a
ncept
nany
plex

hold

e of

Q23. What is union in C?

Ans. Unions are also similar to structure data types with a difference in the way the data is stored and retrieved. The unions store values of different types in a single location. The declaration and the usage of union is same as structure. Union hold only one value for one datatype. If a new value is assigned, the previous value is automatically erased. The unions are declared as :

```
union<user_defined_name>
{       field_member1;
        field_member2;
        -
        -
};
```

In above declaration

The keyword **union** is used to declare the union data type.

The **user_defined_name** is name assigned to the union by user.

This is followed by the curly braces which includes various members of the union.

From the above declaration, the general format of the union can be given as:

```
<storage_class>union<user_define_name>
{       datatype fieldmembers1;
        datatype fieldmembers2;
        datatype fieldmembers3;
        -
        -
};
```

Where the **storage_class** is optional. The keyword union and curly braces are essential and the data / fieldmembers can be any valid C data objects such as **short, int, float** and **char.**

Q24. How can we declare variable in union?

Ans. The variable declaration format in the unions is given same as:

```
union<userdefinename>
{     datatype fieldmembers1;
      datatype fieldmembers2;
      -
      -
} variable1, variable2,....., variable;
```

For Example:

```
union data
{     int a;
      float b;
      char c;
}x, y, z;
```

Here, in above example x, y and z are the variables of the union having data size similar to data. Processing with Unions the various field members of the union can be accessed using a dot operator between the union variable name and the field member name.

Q25. What is the difference between array and structure?

Ans. Both arrays and structures are classified as structured data type, as they provide a mechanism that enable us to access and manipulate data in a relatively easier manner. But they differ is a number of ways:

1. An array is a collection of related data elements of some type structure can have elements of different types.
2. An array is derived data type where as a structure is a programmer defined.
3. An array behaves like a built in data type. All we have to do is to declare an array variable and use it. But in the case of a structure, first we have to design and declare a data structure before the variables of that type are declared and used.

Q26. What is the difference between structure and union?

Ans. Structure is a collection of heterogeneous type of data i.e. different types of data. The various individual data components in a structure can be accessed and processed separately. A structure is a collection of variables referenced under one name, providing a convenient means of keeping related information together. A structure declaration forms a template that may be used to create structure objects (that is, instances of a structure).

Unions are also similar to structure data types with a difference in the way the data is stored and retrieved. The unions store values of different types in a single location. The declaration and the usage of unions is same as structures. Unions hold only one value for one datatype. If a new value is assigned, the previous value is automatically erased. The unions are declared as :

```
union<user_defined_name>
{     field_members1;
      field_members2;
      -
      -
};
```

In above declaration

The keyword **union** is used to declare the union data type.