

# Programming Fundamentals

# Introduction to Structure

- Problem:
  - How to group together a collection of data items of different types that are logically related to a particular entity???
  - (~~Array~~) NO
- *Solution:* **Structure**

# Structure

- **Arrays** allow to define type of variables that can hold several data items of the same kind.
- **structure** is user defined data type that allows to combine data items of different kinds.

# Structure

- A structure is a collection of variables of different data types under a single name.
- The variables are called **members** of the structure.
- The structure is also called a user-defined data type.

- **Structures** are used to represent a record. Suppose you want to keep track of your books in a library. You might want to track the following attributes about each book

- Title
- Author
- Subject
- Book ID

# Defining a structure

- **struct** keyword is used to define a structure.  
    **struct** defines a new data type which is a collection of primary and derived data types

Syntax:

```
struct structure_name  
{  
    // structure member 1  
    //structure member 2  
    //structure member 3  
    .....  
};
```

# Example

```
struct employee
```

```
{  
    int eid;  
    char name[30];  
    char city[20];  
    float salary;  
};
```

```
struct student
```

```
{  
    int rollno;  
    char  
    name[30];  
    char  
    city[20];  
    float marks;  
};
```

# Create struct variables

- When a struct type is declared, no storage or memory is allocated.
- To allocate memory of a given structure type and work with it, we need to create variables.

```
struct employee
{
    int eid;
    char name[30]; char city[20];

    float salary;
}

;

void main ()
{
    employee e1,e2,e3;
}
```



Another way of creating a structure variable is

```
struct employee
```

```
{
```

```
    int eid;
```

```
    char name[30]; char city[20];
```

```
    float salary;
```

```
}e1,e2,e3;
```

# Accessing members of structure

There are two types of operators used for accessing members of a structure

1. `.` : Member operator
2. `->` : Structure pointer operator

**struct employee**

{

**int empid;**

**char name[30]; char city[20];**

**float salary;**

**}e1,e2,e3;**

Suppose here want to access salary of employee e1 we can do

`e1.salary;`

`e.name;`

`e1.eid;`

- Let us consider we have a structure as:

```
struct student  
{  
  char  
  name[20]; int  
  roll;  
  char  
  remarks;  
  float marks;  
};
```

If we want to keep record of 100 students, we have to make 100 structure variables like st1, st2, ...,st100.

In this situation we can use array of structure to store the records of 100 students which is easier and efficient to handle (because loops can be used).

# ARRAY OF STRUCTURE...

Two ways to declare an array of structure:

```
struct student  
{  
    Char name[20];  
    int roll;  
    Char remarks;  
    float marks;  
}  
st[100];
```

```
struct student  
{  
    char name[20];  
    int roll;  
    char remarks;  
    float marks;  
};  
struct student st[100];
```

# Array of structure

- Declaring an array of structure is same as declaring an array of fundamental types
- Since an array is a collection of elements of the same type. In an array of structures, each element of an array is of the structure type

struct car

```
{  
    char make[20];  
    char model[30]; int value;  
};  
struct car arr_car[10];
```

```
struct car arr_car[10];
```

	make	model	value
arr_car[0]	arr_car[0].make	arr_car[0].model	arr_car[0].value
arr_car[1]	arr_car[1].make	arr_car[1].model	arr_car[1].value
arr_car[2]	arr_car[2].make	arr_car[2].model	arr_car[2].value
⋮	⋮	⋮	⋮
arr_car[9]	arr_car[9].make	arr_car[9].model	arr_car[9].value

An array of structure

# READING VALUES

```
for(i=0; i<5; i++)  
{  
    cout<<" Enter roll number:";  
    cin>>s[i].roll_no;  
  
    cout<<"Enter first name:";  
    gets(s[i].f_name);  
  
    Cout<<"Enter Lastname:";  
    gets(s[i].l_name);  
}
```

# SORTING VALUE S

```
for(i=0; i<5; i++)  
{  
    for(j=i+1; j<5; j++)  
    {  
        if(s[i].roll_no<s[j].roll_no  
        )  
        {  
            temp = s[i].roll_no;  
            s[i].roll_no=s[j].roll_n  
            o; s[j].roll_no=temp;  
        }  
    }  
}
```



# QUESTION

- Define a structure of employee having data members name, address, age and salary. Take the data for n employees in an array and find the average salary.
- Write a program to read the *name*, *address*, and *salary* of 5 employees using array of structure. Display information of each employee in alphabetical order of theirname.

```
#include <iostream>
using namespace std;
struct Rectangle {
    int width, height;
    Rectangle(int w, int h)
    {
        width = w;
        height = h;
    }
    void areaOfRectangle() {
        cout << "Area of Rectangle is: " << (width * height);
    }
};
int main(void) {
    struct Rectangle rec = Rectangle(4, 6);
    rec.areaOfRectangle();
    return 0;
}
```