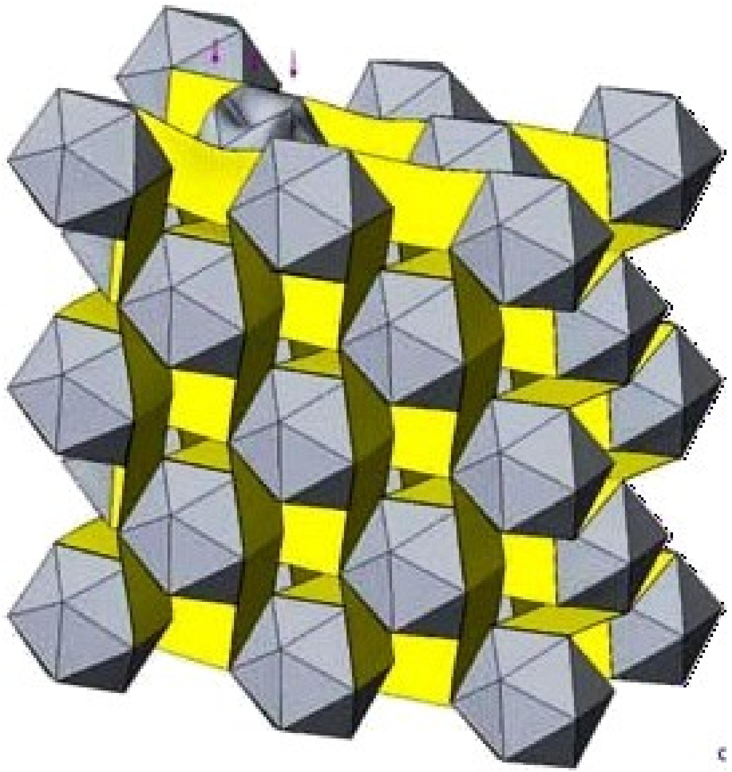
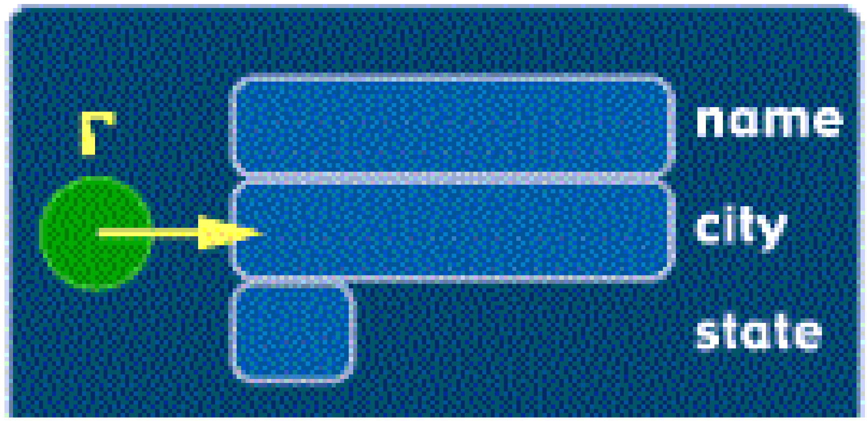




S22_2

Array of Structures & Pointers to Structures

Objectives

- To learn and appreciate the following concept

    - Structures and Functions

    - Pointers and Structures

Session outcome

- At the end of session one will be able to
  - Understand the concept of structures and functions
  - Understand the concept of pointers to structures
  - Write programs on structures using function and pointers

Structures: overview

- Definition & structure variable declaration

```
struct student
  {   int rollno;
      int age;
      char name[20];
      }s1, s2, s3;
```

- Initialization

```
 int main( ){
 struct
  { int rollno;
      int age;
      }stud={20, 21};
      …
      …
      return 0;
}
```

- Giving values to members

Using dot operator '.'

s1. rollno = 25;
cin>>s1.name;

'.' operator acts as Link between member and a Structure variable.

- Assign & compare members

s1 = s2 ;   assignment (allowed)

----

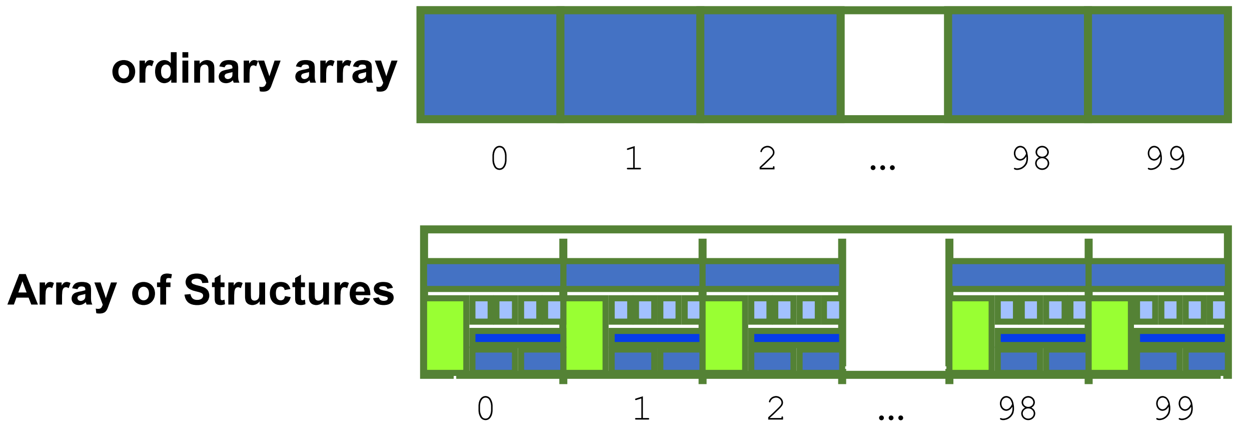
s1 == s2   comparison (not allowed)
s1!=s2   comparison (not allowed)

----

s1.rollno == s2.rollno;   (allowed)
s1.rollno!=s2.rollno;   (allowed)

Previous class

- Array of Structures
- Arrays within Structures
- Structures within Structures



Array within Structures

```
struct marks
{
  int  rollno;
  float  subject[3];
} student[2] ;

student[i].subject[j]
```

```
struct student{
   int rollno;
   char name[15];
   struct {
        int sub1;
        int sub2;
        int sub3;
     }marks;
  }fs[3];

fs[i].marks.sub1
```

Structures and Functions

```
int add(int, int) ;     //function declaration
int main()
{
printf("Enter the value of a : ");
scanf("%d",&sum.a);
printf("\nEnter the value of b : ");
scanf("%d",&sum.b);
sum.c = add(sum. a, sum.b);  //passing structure members as arguments to function
printf("\nThe sum of two value are : ");
printf("%d ", sum.c);
return 0;
}
```

```
//Structure definition
struct addition{
    int a, b;
    int c;
}sum;
```

```
//function definition
int add(int x, int y)
{
int sum1;
sum1 = x + y;
return(sum1);
}
```

Output:

```
Enter the value of a 10
Enter the value of b 20
The sum of two values 30
```

Structures and Functions

```
//Structure definition
struct student
{
        int id;
        char name[20];
        float percentage;
};
```

```
int main()
{
        struct student record;

        record.id=1;
        strcpy(record.name, "Raju");
        record.percentage = 86.5;
         func(record); // passing entire structure
        return 0;
}
//function definition
void func(struct student record)
{
        printf(" Id is: %d \n", record.id);
        printf(" Name is: %s \n", record.name);
        printf(" Percentage is: %f \n", record.percentage);
}
```

Output:

```
Id is: 1
Name is: Raju
Percentage is: 86.500000
```

Structures and Functions

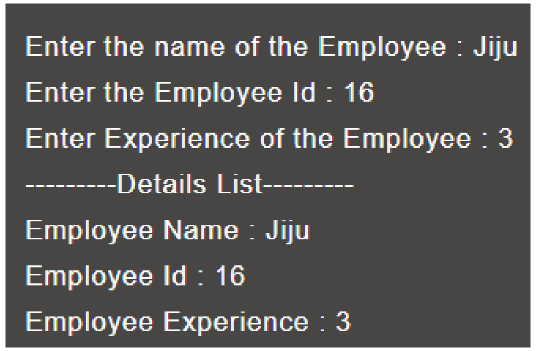
```
//Structure definition
struct employee {
    char name[40];
    int empid;
    int experience;
}emp;
```

```
void displaydetails(struct employee*);
```

```
//function definition
void displaydetails(struct employee *empptr)
{
printf("\n---------Details List--------- \n ");
printf("Employee Name : %s",empptr->name);
printf("\nEmployee ID : %d ",empptr->empid);
printf("\nEmployee Experience : %d ",empptr->experience);
}
```

```
int main() {
struct employee *empptr;   //pointer declaration
empptr = &emp;   //initialization
printf("\nEnter the name of the Employee : ");
scanf("%s", empptr→name);
printf("\nEnter the Employee Id : ");
scanf("%d",&empptr→empid);
printf("\nEnter Experience of the Employee : ");
scanf("%d",&empptr→experience);
displaydetails(empptr); // passing structure using pointers
return 0;
}
```

Output:



```
Enter the name of the Employee : Jiju
Enter the Employee Id : 16
Enter Experience of the Employee : 3
---------Details List---------
Employee Name : Jiju
Employee Id : 16
Employee Experience : 3
```

Go to posts/chat box for the link to the question

submit your solution in next 2 minutes

The session will resume in 3 minutes

Pointers and structures

Consider the following structure

```
struct inventory {
char name[30];
int number;
float price;
} product[2],*ptr;
```

This statement declares product as an array of 2 elements, each of the type struct inventory.

ptr=product; assigns the address of the **zero**th element of **product** to **ptr**

or **ptr** points to **product[0];**

Pointers and Structures

Its members are accessed using the following notation

ptr →name
ptr →number
ptr →price

The symbol → is called arrow operator (also known as member selection operator)

When **ptr is incremented by one**, it points to the next record. i.e. **product[1]**

The member price can also be accessed using

(*ptr).price

Parentheses is required because '.' has higher precedence than the operator *

Pointers and Structures- *example*

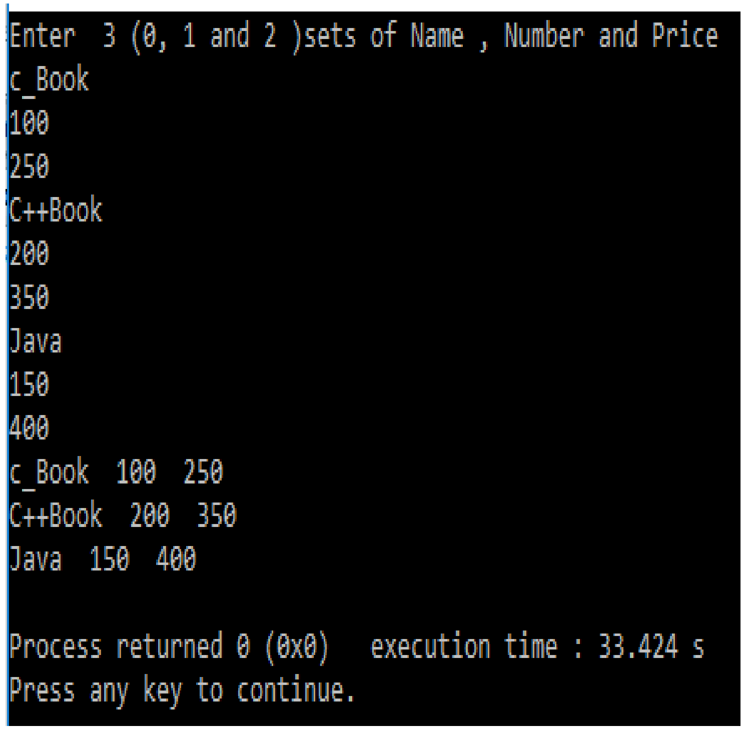
```
struct invent {
 char name[30];
 int number;
 float price;
};
```

```
#include <stdio.h>

int main()
{
  struct invent prod[3], *ptr;
  printf("Enter  3 (0, 1 and 2 )sets of Name,
  Number and Price");

  for(ptr = prod; ptr < prod+3; ptr++)
     scanf("%s %d %f",ptr ->name, &ptr ->number,
              &ptr ->price);
  ptr=prod;

  while(ptr < prod+3)
  {
   printf("%s  %d  %f\n", ptr ->name,
             ptr ->number, ptr ->price);     ptr++;
  }
  return 0;
}
```

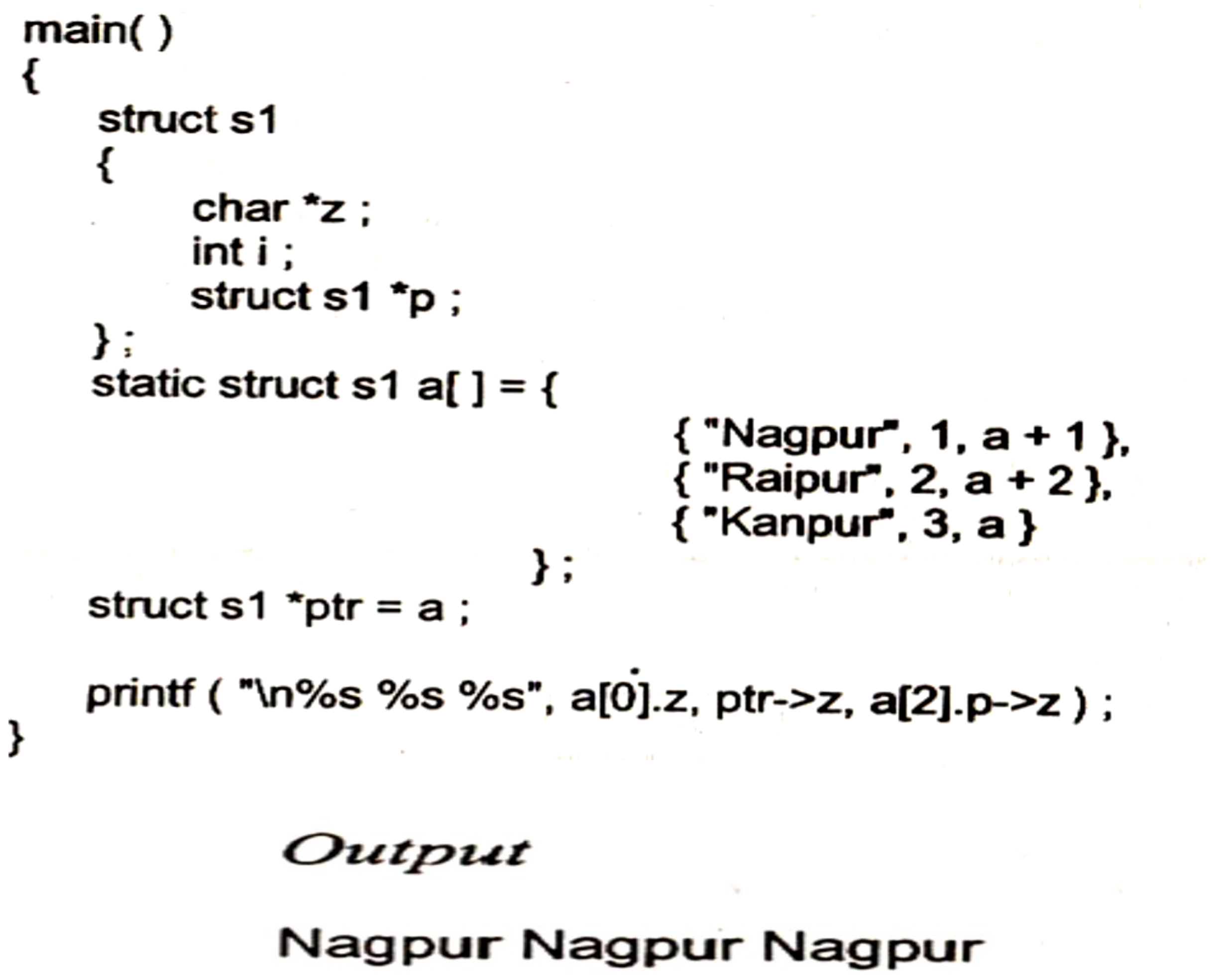


```
Enter  3 (0, 1 and 2 )sets of Name , Number and Price
c_Book
100
250
C++Book
200
350
Java
150
400
c_Book  100  250
C++Book  200  350
Java  150  400

Process returned 0 (0x0)   execution time : 33.424 s
Press any key to continue.
```

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

Pointers and Structures- example



```
main( )
{
    struct s1
    {
        char *z ;
        int i ;
        struct s1 *p ;
    } ;
    static struct s1 a[ ] = {
                                { "Nagpur", 1, a + 1 },
                                { "Raipur", 2, a + 2 },
                                { "Kanpur", 3, a }
                            } ;
    struct s1 *ptr = a ;

    printf ( "\n%s %s %s", a[0].z, ptr->z, a[2].p->z ) ;
}
```

Output

```
Nagpur Nagpur Nagpur
```

Pointers and Structures- example

```
main( )
{
    struct a
    {
        char ch[7] ;
        char *str ;
    } ;

    struct b
    {
        char *c ;
        struct a ss1 ;
    } ;

    struct b s2 = { "Raipur", "Kanpur", "Jaipur" } ;

    printf ( "\n%s %s", s2.c, s2.ss1.str ) ;
    printf ( "\n%s %s", ++s2.c, ++s2.ss1.str ) ;
}
```

Output

```
Raipur Jaipur
aipur aipur
```

Summary

- Structures and Functions
- Pointers and Structures