

Batch:A3

Roll No.: 16010421075

Experiment No.: 1

Aim: Implement and Demonstrate the use of arrays, array of structure and pointers using C.

**Resources needed:** Turbo C/C++ editor and C compiler (Online/Offline)

Theory

1. **Arrays**

An array in C/C++ or be it in any programming language is a collection of similar data items stored at contiguous memory locations and elements can be accessed randomly using indices of an array.

Examples

#include <iostream>

using namespace std;

int main()

{

**int** arr1[10];

**int** n = 10;

**int** arr2[n];

**return** 0;

}

1. **Structures**

A structure is a key word that create user defined data type in C/C++. A structure

creates a data type that can be used to group items of possibly different types into a single type.

Examples

struct address

{

**char** name[50];

**char** street[100];

**char** city[50];

**char** state[20];



**int** pin;

};

1. **Array of Structure**

An array of structres in C can be defined as the collection of multiple structures variables where each variable contains information about different entities.

Examples

#include<stdio.h>

struct Employee{ int id;

int salary;

};

int main()

{

printf("How many employees do you want to enter: "); int n;

scanf("%d",&n);

struct Employee E[n];

for(int i=0;i<n;i++){ int x;

printf("Enter Employee ID: "); scanf("%d",&x);

E[1].id=x;

printf("Enter Salary: "); scanf("%d",&x);

E[1].salary=x;

}

for(int i=0;i<n;i++){

printf("%d - %dRs\n",E[i].id,E[1].salary);

}

}

1. **Pointers and Pointers to Structures**

Structure Pointer: It is defined as the pointer which points to the address of the memory block that stores a structure is known as the structure pointer

Examples

#include<stdio.h>

struct Employee{

char name[50]; int salary;

};

int main()

{

struct Employee \*E;

int x;

printf("Enter Employee name: "); scanf("%s",E->name);

printf("Enter Salary: "); scanf("%d",&E->salary);

printf("%s - %d",E->name,E->salary);

}

1. **Function**

A function is a block of code which only runs when it is called. You can pass data, known as parameters, into a function.

Functions are used to perform certain actions, and they are important for reusing code: Define the code once, and use it many times.

Examples

#include<stdio.h>

void swap(int \*a,int \*b); int main()

{

int a,b;

printf("Enter two number: "); scanf("%d%d",&a,&b);

printf("Before swap- a=%d b=%d",a,b); swap(&a,&b);

printf("\nAfter swap- a=%d b=%d",a,b);

}

void swap(int \*a,int \*b)

{

\*a=\*a+\*b;

\*b=\*a-\*b;

\*a=\*a-\*b;

}

Activity :

Program should demonstrate the use of concepts of arrays, pointers, structures, array of

structure, pointers to structure. Students are required to choose a proper example and show the use of above concept in the implementation of the example. Consider implementing a modular programming technique by making use of user defined functions.

**Results:** A C program depicting the correct behaviour of mentioned concept and capable of handling all possible exceptional conditions/inputs and the same is reflecting clearly in the output.

Program and Output : Array

#include<stdio.h>

int main()

{

printf("How many elements you want to enter: "); int n;

scanf("%d",&n); int arr[n];

int i;

for(i=0;i<n;i++){ int x;

scanf("%d",&x); arr[i]=x;

}

for(i=0;i<n;i++){

printf("%d ",arr[i]);

}

return 0;

}

Structures

#include<stdio.h>

struct Employee{ int id;

int salary;

};

int main()

{

struct Employee E1;

int x;

printf("Enter Employee ID: "); scanf("%d",&x);

E1.id=x;

printf("Enter Salary: "); scanf("%d",&x);

E1.salary=x;

printf("%d - %dRs",E1.id,E1.salary);

}

Array Of Structure

#include<stdio.h>

**struct Employee**{ **int** id;

**int** salary;

};

int main()

{

printf("How many employees do you want to enter: ");

**int** n;

scanf("%d",&n);

struct Employee E[n];

**for**(**int** i=0;i<n;i++){

**int** x;

printf("Enter Employee ID: "); scanf("%d",&x);

E[1].id=x;

printf("Enter Salary: "); scanf("%d",&x);

E[1].salary=x;

}

**for**(**int** i=0;i<n;i++){

printf("%d - %dRs\n",E[i].id,E[1].salary);

}

}

Pointer and Pointer to Structures

#include<stdio.h>

struct Employee{

char name[50]; int salary;

};

int main()

{

struct Employee \*E;

int x;

printf("Enter Employee name: "); scanf("%s",E->name);

printf("Enter Salary: "); scanf("%d",&E->salary);

printf("%s - %d",E->name,E->salary);

}

Function

#include<stdio.h>

void swap(int \*a,int \*b); int main()

{

int a,b;

printf("Enter two number: "); scanf("%d%d",&a,&b);

printf("Before swap- a=%d b=%d",a,b); swap(&a,&b);

printf("\nAfter swap- a=%d b=%d",a,b);

}

void swap(int \*a,int \*b)

{

\*a=\*a+\*b;

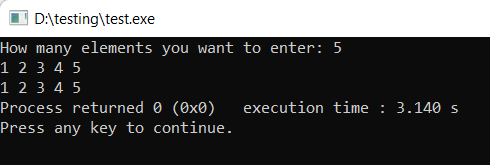
\*b=\*a-\*b;

\*a=\*a-\*b;

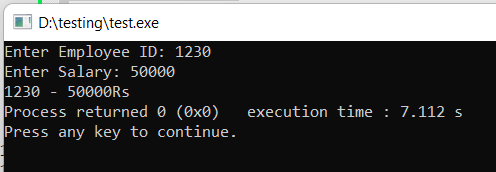
}

Outcomes:

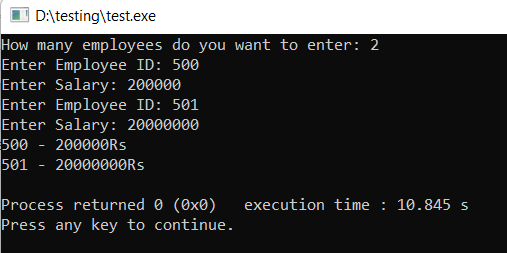
Array



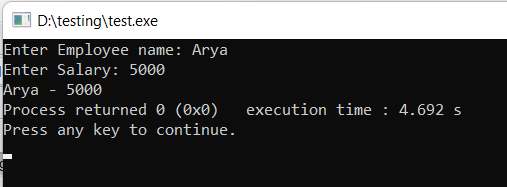
Structure



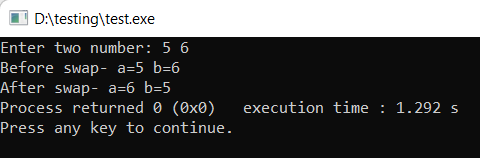
Array of Structure



Pointer to Structure



Function



Conclusion:

Successfully implemented the concepts of Arrays, Pointers, structures and Functions

Grade: AA / AB / BB / BC / CC / CD /DD Signature of faculty in-charge with date

References:

Books/ Journals/ Websites:

* Y. Langsam, M. Augenstin and A. Tannenbaum, “**Data Structures using C**”, Pearson Education Asia, 1st Edition, 2002
* **Data Structures A Psedocode Approach with C**, Richard F. Gilberg&Behrouz A. Forouzan, secondedition, CENGAGE Learning