

Lab 08

Objectives

After performing this lab, students will be able to

- Understand structures and use them in programs
- Understand pointers and create programs using them

Exercises

1. Write a C++ program that maintain record of students. Student contains following details:

- a. ID
- b. Name
- c. Department
- d. Email
- e. Phone number

Create a structure names **Student**. Ask the user to enter record for 5 students and store those details in variables of **Student** type. Finally, print those records on screen.

2. Write a C++ program to do:
 - a. Create a structure/record to store products; each product has a name, a model number and a price.
 - b. Choose appropriate types for these fields/attributes.
 - c. Your program should contain a loop which allows entry via the keyboard of up to 10 products, and stores them in an array
 - d. Finally, the program should include a function to display all the products details after the entry loop has finished.
3. Write a C++ program that compute Net Salary of Employee. Program should contain two user defined functions **empSalary()** and **display()**.
 - a. Create a structure of Employee that contains following data members:
 - i. EmployeeNumber, Name, BasicSalary, HouseAllowance, MedicalAllowance, Tax, GrossPay and NetSalary
 - b. EmployeeNumber, Name, and BasicSalary should be taken input from the user
 - c. The function **empSalary()** should compute the Employee salary with given criteria
 - i. HouseAllowence = 10% of BasicSalary
 - ii. Medical Allowence = 5% of Basic Salary
 - iii. Tax = 4 % of Basic Salary
 - iv. GrossSalary = Basic+HouseAllowence+MedicalAllowence
 - v. NetSalary = GrossSalary – Tax

- d. The function `display()` should display the details of Employee, like shown below:

```
#Sample Output

Enter the Employee Number :10129
Enter the employee name: Ahmed Ali
Enter the Basic Salary: 16500

*****

      EMPLOYERS SALARY DETAILS
*****

Employee Number:10129
Employee Name: Ahmed Ali
Basic Salary: 16500
House Allowance:1650
Medical Allowance: 825
Gross Salary: 18975
Tax Deduction: 660
Net Salary: 18315
```

4. Initialize an integer array of 5 elements. Then, print values of all elements along with their addresses using pointers.
5. Write a program that should have a user-defined function names **`bubbleSort()`**, which should have one parameter of pointer to an integer array. The function should sort the values of that array in ascending order (smaller to larger). Then, in the `main()` function, create an integer array initialized with some values within the code and print the array values along with a message saying that is the original/unsorted array values. After that,

pass that array to the function **bubbleSort()** that should sort its values. Finally, print the array values along with a message saying that these are the sorted values.

6. Write a C++ program that creates an integer array of size 10.
 - a. Then, get the array values as input from the user but save them in the array using a pointer.
 - b. Then, print those values using the pointer
 - c. Then, ask the user to search any value and perform a search operation using the pointer
7. Create a function named **swap()** that should swap two integer values (means, exchanges the values between two variables; the value of first variable should be saved in the second one, and the value of the second variable should be saved in the first one, and so on) using pointers. The function parameters should be two pointers. Then, in the **main()** function, create (and initialize) two variables and print their values before calling the **swap()** function. After that, call the swap function. And finally, print those values after the function call to show that the values are swapped/exchanged.
8. Consider the following code:

```
#include <string>
#include <iostream>
using namespace std;

struct Student {
    string name;
    int id;
    int mark[3];
};

void inputStudent(Student* ptr); //function prototype for getting input

// some other function for printing the details

//***** Main Function *****//
int main () {
    Student stu;           // declaring an Student object
    Student* stuPtr = &stu; // defining a pointer for the object

    inputStudent(&stu); // calling function to get input into the object

    // calling the other function to print the details of the object

    return 0;
} // end main
```

In above given code there is a prototype, and a call to a function to input the values into a Student instance. Note that the parameter to this function is the address of a structure instance.

- a) Write the definition for this function at the bottom of the file.
- b) Then, write a function that takes a pointer to an instance of the Student structure and displays the contents

9. Making and Using Dynamically Allocated Arrays.

Let us say that you want to choose how many marks to enter for the student and have the array change size accordingly. This gives you the opportunity to work with dynamic pointers.

You will have to make the following changes to the code previously given:

- a) Change **mark** within the **Student** struct to an integer pointer.
- b) Ask the user from main how many marks he/she would like to enter
- c) Change both the functions to have an additional parameter which is the number of marks.
- d) For the **inputStudent** function, dynamically allocate the marks according to the number required