CL1002 Programming Fundamentals

LAB 11

Structures and Nested Structure

NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

Learning Objectives

1. Structures
2. Nested Structures

1. Structures

Structures are derived data types—they're constructed using objects of other types. Normally, we use structure to store the record or the details of any item or entity. Structure members can be variables of the primitive data types (e.g., int, float, etc.), or aggregates, such as arrays and other structures.

- Keyword struct introduces a structure definition
- The identifier Chocolate is the structure tag, which names the structure definition and is used with struct to declare variables of the structure type—e.g., struct Chocolate kitkat, Mars, Jubilee.
- Variables declared within the braces of the structure definition are the structure's members.
- Members of the same structure type must have unique names, but two different structure types may contain members of the same name without conflict.

1.1 Declaration of Struct

```
struct Chocolate{
        char Name[20];
        float Weight;
        int Calories;
        float Price;
        char ExpiryDate[10];
};
```

1.2 Declaration & Initialization of Struct type Variables

You can declare the variables before the semi-colon(;) or using a proper declaration syntax like other variables in main();

```
struct Chocolate{
        char Name[20];
        float Weight;
        int Calories;
        float Price;
        char ExpiryDate[10];
}var1, var2,var3;
```

```
Int main()
{
Struct Chocolate Kitkat, Mars, Jubilee, mychocolate[3];
//        OR
struct Chocolate myChocolate;
        gets(myChocolate.Name);
        myChocolate.Weight= 20;
        myChocolate.Calories= 500;
        myChocolate.Price= 100;
        strcpy(myChocolate.ExpiryDate,"01-Feb-2021");

//        OR

struct Chocolate Jubilee = {"Jubilee",20.50,500,100,"01-Feb-2021"};

}
```

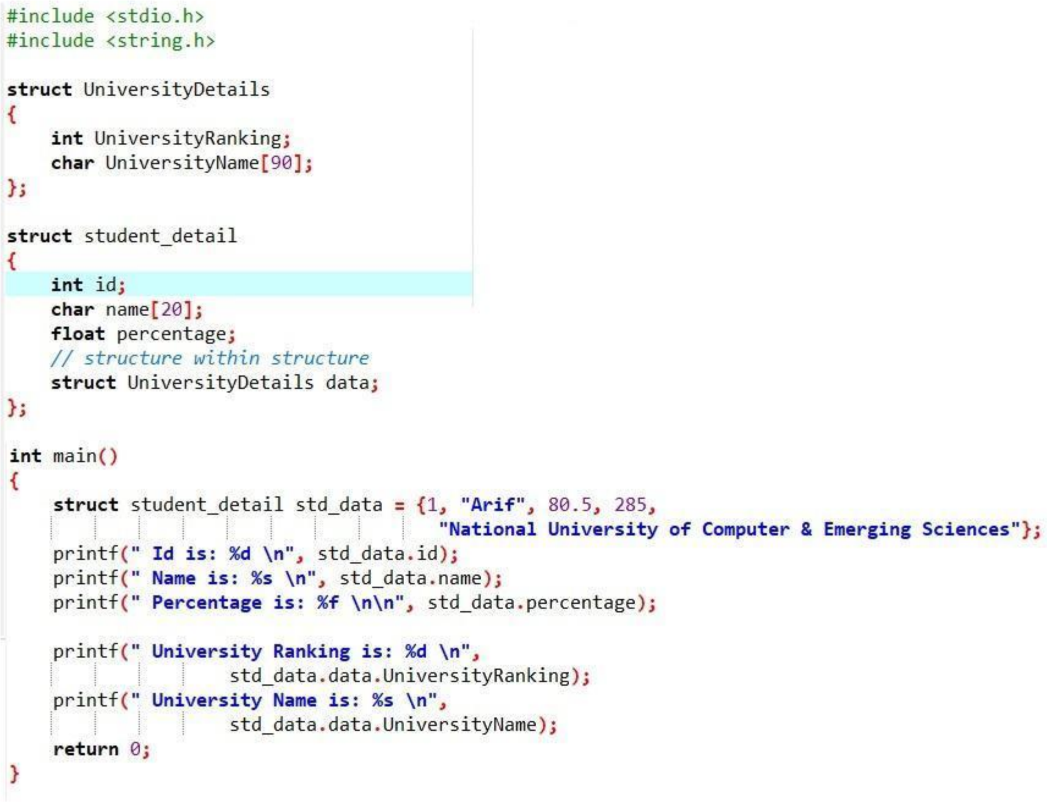
1.3 Declaration & Initialization of Struct type Array

```
int main() {

  struct Chocolate myChocolate[3]; // Array of struct



  int i = 0;

  char buffer[100]; // Buffer for input



  while (i < 3) {

    printf("Enter name: ");

    fgets(buffer, sizeof(buffer), stdin);

    sscanf(buffer, "%s", myChocolate[i].Name);

    printf("Enter weight: ");

    fgets(buffer, sizeof(buffer), stdin);

    sscanf(buffer, "%f", &myChocolate[i].Weight);



    printf("Enter calories: ");

    fgets(buffer, sizeof(buffer), stdin);

    sscanf(buffer, "%d", &myChocolate[i].Calories);
```

```
        printf("Enter price: ");
        fgets(buffer, sizeof(buffer), stdin);
        sscanf(buffer, "%f", &myChocolate[i].Price);

        printf("Enter expiry date: ");
        fgets(buffer, sizeof(buffer), stdin);
        sscanf(buffer, "%s", myChocolate[i].ExpiryDate);

        ++i;
    }

    // To print the array of structs
    i = 0;
    while (i < 3) {
        printf("Name: %s\n", myChocolate[i].Name);
        printf("Weight: %.2f\n", myChocolate[i].Weight);
        printf("Calories: %d\n", myChocolate[i].Calories);
        printf("Price: %.2f\n", myChocolate[i].Price);
        printf("Expiry Date: %s\n", myChocolate[i].ExpiryDate);

        ++i;
    }

    return 0;
}
```

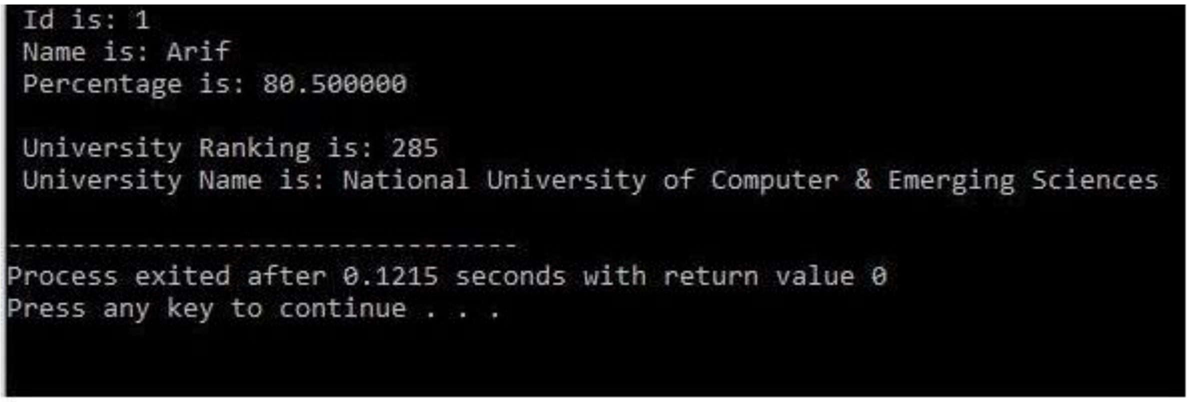
2.0 Nested Structures

Nested structure in C is nothing but structure within structure. One structure can be declared inside another structure as we declare structure members inside a structure. The structure variables can be a normal structure variable, array or a pointer variable to access the data. You can learn the concepts below in this section.



```c
#include <stdio.h>
#include <string.h>

struct UniversityDetails
{
    int UniversityRanking;
    char UniversityName[90];
};

struct student_detail
{
    int id;
    char name[20];
    float percentage;
    // structure within structure
    struct UniversityDetails data;
};

int main()
{
    struct student_detail std_data = {1, "Arif", 80.5, 285,
                                      "National University of Computer & Emerging Sciences"};
    printf(" Id is: %d \n", std_data.id);
    printf(" Name is: %s \n", std_data.name);
    printf(" Percentage is: %f \n\n", std_data.percentage);

    printf(" University Ranking is: %d \n",
                std_data.data.UniversityRanking);
    printf(" University Name is: %s \n",
                std_data.data.UniversityName);
    return 0;
}
```

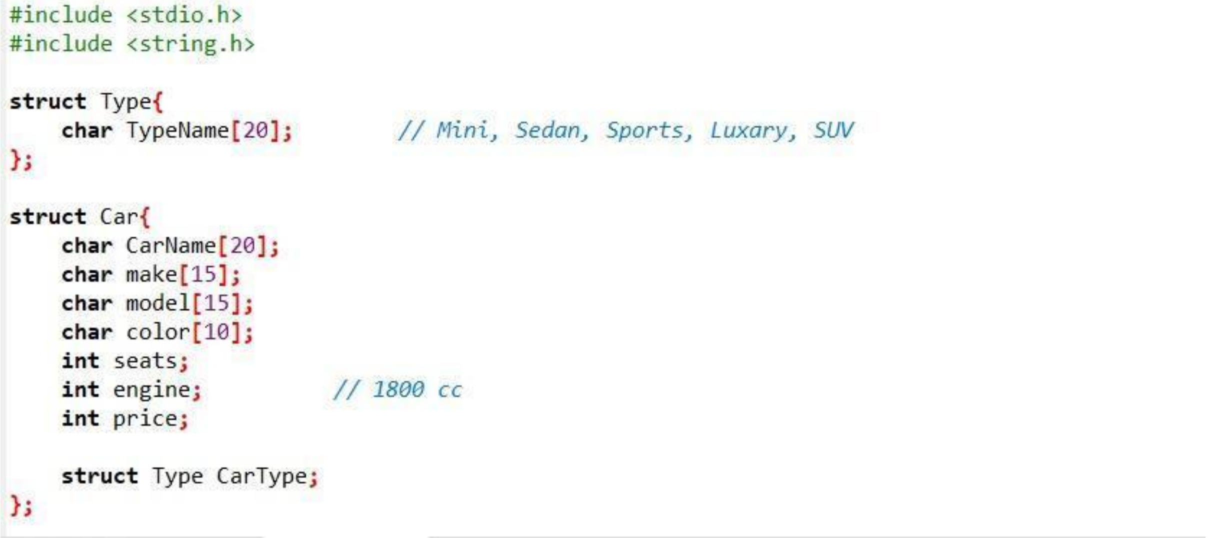
OUTPUT:



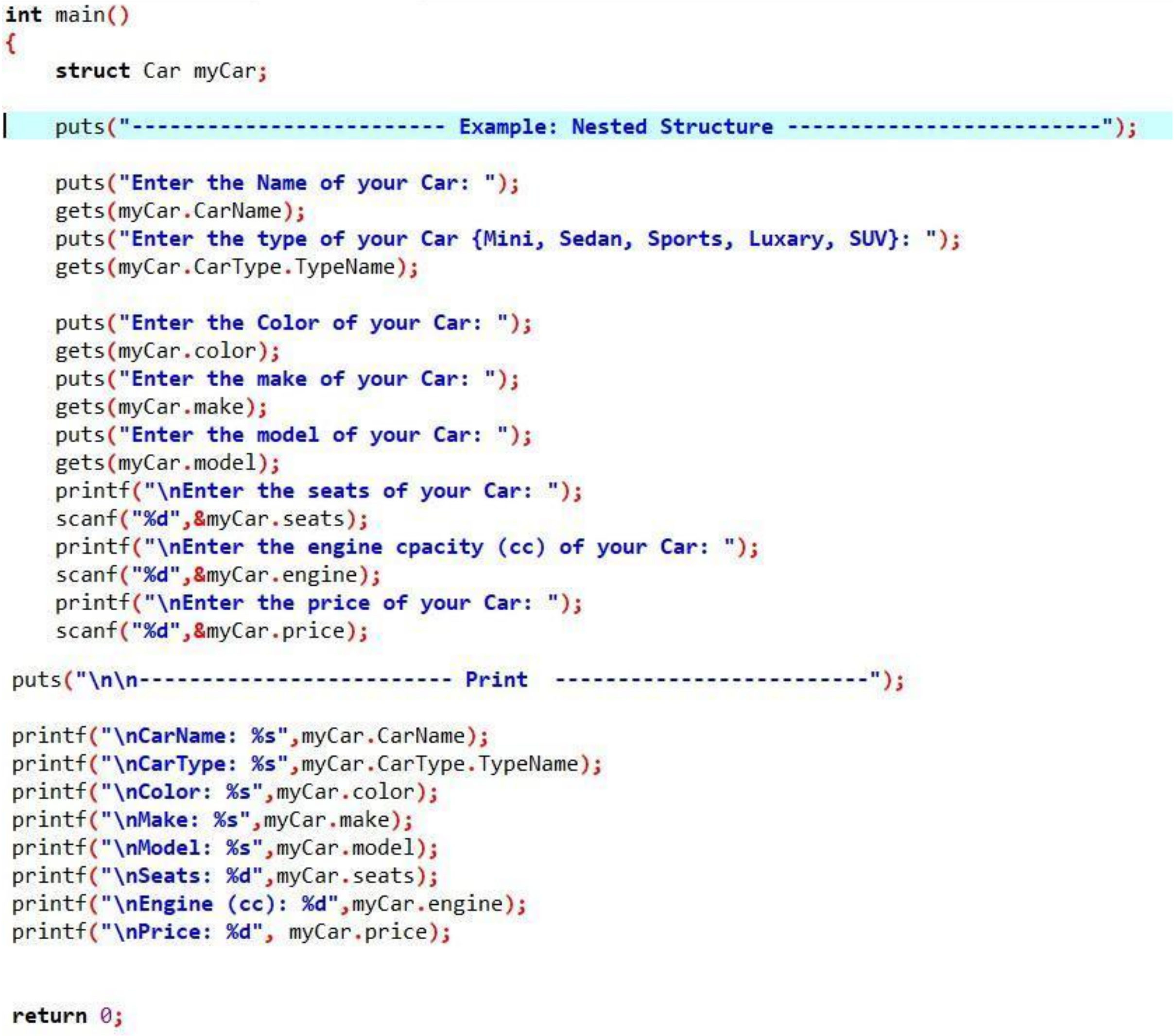
```
 Id is: 1
 Name is: Arif
 Percentage is: 80.500000

 University Ranking is: 285
 University Name is: National University of Computer & Emerging Sciences

--------------------------------
Process exited after 0.1215 seconds with return value 0
Press any key to continue . . .
```

Another example of Nested Structure:

Sample Code:

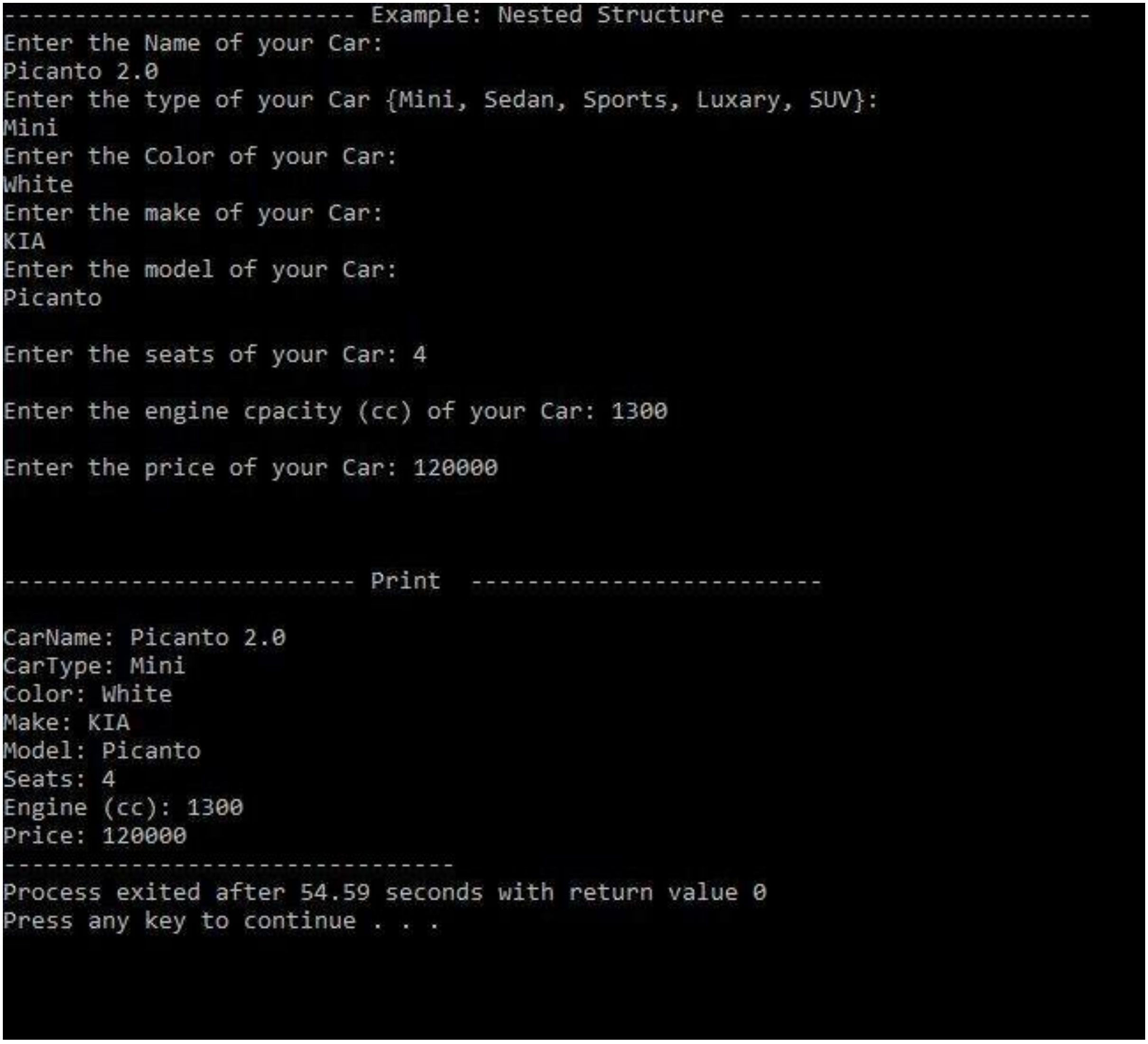


```
#include <stdio.h>
#include <string.h>

struct Type{
    char TypeName[20];          // Mini, Sedan, Sports, Luxary, SUV
};

struct Car{
    char CarName[20];
    char make[15];
    char model[15];
    char color[10];
    int seats;
    int engine;           // 1800 cc
    int price;

    struct Type CarType;
};
```



```
int main()
{
    struct Car myCar;

    puts("------------------------- Example: Nested Structure -------------------------");

    puts("Enter the Name of your Car: ");
    gets(myCar.CarName);
    puts("Enter the type of your Car {Mini, Sedan, Sports, Luxary, SUV}: ");
    gets(myCar.CarType.TypeName);

    puts("Enter the Color of your Car: ");
    gets(myCar.color);
    puts("Enter the make of your Car: ");
    gets(myCar.make);
    puts("Enter the model of your Car: ");
    gets(myCar.model);
    printf("\nEnter the seats of your Car: ");
    scanf("%d",&myCar.seats);
    printf("\nEnter the engine cpacity (cc) of your Car: ");
    scanf("%d",&myCar.engine);
    printf("\nEnter the price of your Car: ");
    scanf("%d",&myCar.price);

puts("\n\n------------------------- Print  -------------------------");

printf("\nCarName: %s",myCar.CarName);
printf("\nCarType: %s",myCar.CarType.TypeName);
printf("\nColor: %s",myCar.color);
printf("\nMake: %s",myCar.make);
printf("\nModel: %s",myCar.model);
printf("\nSeats: %d",myCar.seats);
printf("\nEngine (cc): %d",myCar.engine);
printf("\nPrice: %d", myCar.price);


return 0;
```

OUTPUT:



```
------------------------- Example: Nested Structure -------------------------
Enter the Name of your Car:
Picanto 2.0
Enter the type of your Car {Mini, Sedan, Sports, Luxary, SUV}:
Mini
Enter the Color of your Car:
White
Enter the make of your Car:
KIA
Enter the model of your Car:
Picanto

Enter the seats of your Car: 4

Enter the engine cpacity (cc) of your Car: 1300

Enter the price of your Car: 120000



------------------------- Print  -------------------------

CarName: Picanto 2.0
CarType: Mini
Color: White
Make: KIA
Model: Picanto
Seats: 4
Engine (cc): 1300
Price: 120000
--------------------------------
Process exited after 54.59 seconds with return value 0
Press any key to continue . . .
```

3.0 Pointers in Structure:

```
#include <stdio.h>
struct Student{
        int id;
        char name[15]
};
int main(){
    struct Student s1, *ptr;
s1.id = 2;
gets(s1.name);
ptr = &s1;
//printf("%d",*(ptr.id);   // generates the errors
//printf("%d",*(ptr.name); // generates the errors
printf("%d ",(*ptr).id);
printf("%s ",(*ptr).name);
printf("\n%d ",ptr->id);
printf("%s ",ptr->name); }
```

Exercises:

1. Create a structure to specify data on students given below:

Roll number, Name, Department, Course, Year of joining Assume that there are not more than 450 students in the college.

- Print names of all students who joined in a particular year.
- Print the data of a student whose roll number is given.

2. An automobile company has a serial number for engine parts starting from AA0 to FF9. The other characteristics of parts to be specified in a structure are: Year of manufacture, material and quantity manufactured.

- Specify a structure to store information corresponding to a part.
- Write a program to retrieve information on parts with serial numbers between BB1 and CC6.

3. Write a program to compare two dates entered by the user. Make a structure named Date to store the elements day, month and year to store the dates. If the dates are equal, display "Dates are equal" otherwise display "Dates are not equal".

4. Write a structure to store the names, salary and hours of work per day of 10 employees in a company. Write a program to increase the salary depending on the number of hours of work per day as follows and then print the name of all the employees along with their final salaries.

● Hours of work per day	8	10	>=12
● Increase in salary	$50	$100	$150

5. Consider there are two structures: Employee (dependent structure) and another structure called Organization (Outer structure). The structure Organization has the data members like organisation_name,organisation_number. The Employee structure is nested inside the structure Organization and it has the data members like employee_id, name, salary.

```
org.emp.employee_id;
org.emp.name;
org.emp.salary;
```

```
org.organisation_name;
org.organisation_number;
```

Here, org is the structure variable of the outer structure Organisation and emp is the structure variable of the inner structure Employee.

Output the following data using above structure

```
The size of structure organisation : 123
Organisation Name : NU-Fast
Organisation Number : NUFAST123ABC
```

```
Employee id : 127
Employee name : Linus Sebastian
Employee Salary : 400000
```

6. Create a structure named Date having day, month and year as its elements. Store the current date in the structure. Now add 45 days to the current date and display the final date.

7. Let us work on the menu of a library. Create a structure containing book information like accession number, name of author, book title and flag to know whether book is issued or not.

Create a menu in which the following can be done.

1 - Display book information

2 - Add a new book

3 - Display all the books in the library of a particular author

4 - Display the number of books of a particular title

5 - Display the total number of books in the library

6 - Issue a book

(If we issue a book, then its number gets decreased by 1 and if we add a book, its number gets increased by 1)

8. You are transporting some boxes through a tunnel, where each box is a parallelepiped, and is characterized by its length, width and height.

The height of the tunnel is 41 feet, and the width can be assumed to be infinite. A box can be carried through the tunnel only if its height is strictly less than the tunnel's height. Find the volume of each box that can be successfully transported to the other end of the tunnel. Note: Boxes cannot be rotated.

Sample Input 0

```
4
5 5 5
1 2 40
10 5 41
7 2 42
```

Sample Output 0

```
125
80
```

Explanation: The first box is low, only 5 feet tall, so it can pass through the tunnel and its volume is 5*5*5=125. The second box is sufficiently low, its volume is 1*2*40=80. The third box is exactly 41 feet tall so it cannot pass. The same can be said about the fourth box.

Note: Only use structs for this question

9. You need to implement the following 2 struct.

```
struct Student{};  struct Register{};
```

Student contains attributes StudentId, FirstName, LastName, cellno, email.

Register contains the attributes CourseId, CourseName.

Now you need to inherit the Register struct in Student struct. It means that the student struct holds the variable of Register struct variable. After that you need to take input for 5 students and then print them

[Hint: Declare array of struct Student std[5]; for 5 students]