LAB-4

Structure Operations in C

Q1) Write a program to swap two complex numbers using a struct.

```
#include <stdio.h>

struct complex{
   float real;
   float  img;
};

int main() {
        // your code goes here
        struct complex num1,num2;
        printf("Complex number to swap\n");
        printf("Enter 1st Complex number First Real part then Imaginary
part\n");
        scanf("%f %f",&num1.real,&num1.img);
        printf("Num1: %.2f + %.2fi\n", num1.real, num1.img);
        printf("Enter 2nd Complex number First Real part then Imaginary
part\n");
        scanf("%f %f",&num2.real,&num2.img);
        printf("Num2: %.2f + %.2fi\n", num2.real, num2.img);
        float tempR=num1.real;
        num1.real=num2.real;
        num2.real=tempR;
        float tempI=num1.img;
        num1.img=num2.img;
        num2.img=tempI;
        printf("After Swapping : \n");
        printf("Num1: %.2f + %.2fi\n", num1.real, num1.img);
        printf("Num2: %.2f + %.2fi\n", num2.real, num2.img);
        return 0;
}
```

Sample Input

```
20
```

34

6

12

Your Output

```
Complex number to swap
Enter 1st Complex number First Real part then Imaginary part
Num1: 20.00 + 34.00i
Enter 2nd Complex number First Real part then Imaginary part
Num2: 6.00 + 12.00i
After Swapping :
Num1: 6.00 + 12.00i
Num2: 20.00 + 34.00i
```

Q2) Write a program to store and display information of n employees using an array of structures.

```
#include <stdio.h>

struct Employee {
   int id;
   char name[50];
   float salary;
};

int main() {
   int n, i;
   scanf("%d", &n);
   printf("Enter number of employees: %d",n);

   struct Employee emp[n];

   for(i = 0; i < n; i++) {
      printf("\nEnter details of Employee %d\n", i+1);
      printf("Enter ID: ");
      scanf("%d", &emp[i].id);
```

```
    printf("Enter Name: ");
    scanf("%s", emp[i].name);
    printf("Enter Salary: ");
    scanf("%f", &emp[i].salary);
  }

  printf("\nAfter storing, Employee Information is:\n");
  for(i = 0; i < n; i++) {
    printf("\nEmployee %d\n", i+1);
    printf("ID: %d\n", emp[i].id);
    printf("Name: %s\n", emp[i].name);
    printf("Salary: %.2f\n", emp[i].salary);
  }

  return 0;
}
```

Sample Input

```
2
34
Ohi
98000
20
Rav
92000
```

Your Output

```
Enter number of employees: 2
Enter details of Employee 1
Enter ID: Enter Name: Enter Salary:
Enter details of Employee 2
Enter ID: Enter Name: Enter Salary:
After storing, Employee Information is:
Employee 1
ID: 34
Name: Ohi
Salary: 98000.00
```

```
Employee 2
ID: 20
Name: Rav
Salary: 92000.00
```

Q3) Explain how nested structures work. Write a program where a struct for date is nested inside a struct for student.

```
#include <stdio.h>

struct Address {
   char city[50];
   int pincode;
};

struct Student {
   char name[50];
   int class;
   struct Address addr;
};

int main() {
   int n, i;
   printf("Enter number of Students: ");
   scanf("%d", &n);

   struct Student Std[n];

   for(i = 0; i < n; i++) {
      printf("\nEnter details of Student %d\n", i+1);
      printf("Enter Name: ");
      scanf("%s", Std[i].name);
      printf("Enter Class: ");
      scanf("%d", &Std[i].class);
      printf("Enter City: ");
      scanf("%s", Std[i].addr.city);
      printf("Enter Pincode: ");
      scanf("%d", &Std[i].addr.pincode);
```

```
  }

  printf("\nStudents Information is:\n");
  for(i = 0; i < n; i++) {
    printf("\nStudent %d\n", i+1);
    printf("Name: %s\n", Std[i].name);
    printf("Class: %d\n", Std[i].class);
    printf("City: %s\n", Std[i].addr.city);
    printf("Pincode: %d\n", Std[i].addr.pincode);
  }

  return 0;
}
```

Sample Input

```
2
Ohi
12
Haze
000
Rav
12
Mist
404
```

Your Output

```
Enter number of Students:
Enter details of Student 1
Enter Name: Enter Class: Enter City: Enter Pincode:
Enter details of Student 2
Enter Name: Enter Class: Enter City: Enter Pincode:
Students Information is:
Student 1
Name: Ohi
Class: 12
City: Haze
```

Pincode: 0

Student 2

Name: Rav

Class: 12

City: Mist

Pincode: 404

Q4) Can a struct contain a pointer to itself? Give an example.

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

struct Node {
   int data;
   struct Node* next;
};

int main() {
   int n, i, val;
   struct Node *head = NULL, *temp = NULL, *prev = NULL;

   printf("Enter number of nodes: \n");
   scanf("%d", &n);

   for(i = 0; i < n; i++) {
      temp = (struct Node*)malloc(sizeof(struct Node));
      printf("Enter data for node %d: \n", i+1);
      scanf("%d", &val);
      temp->data = val;
      temp->next = NULL;

      if(head == NULL){
         head = temp;
      }
      else{
         prev->next = temp;
      }
      prev = temp;
```

```
    }

    temp = head;
    printf("Result of Nodes : ");
    while(temp != NULL) {
        printf("%d -> ", temp->data);
        temp = temp->next;
    }
    printf("NULL\n");

    temp = head;
    while(temp != NULL) {
        struct Node* next = temp->next;
        free(temp);
        temp = next;
    }

    return 0;
}
```

Input & Output

```
Enter number of nodes:
3
Enter data for node 1:
34
Enter data for node 2:
20
Enter data for node 3:
12
Result of Nodes : 34 -> 20 -> 12 -> NULL
```

Q4) Can a struct contain a pointer to itself? Give an example.

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

```
struct Node {
   int data;
   struct Node* next;
};

int main() {
   int n, i, val;
   struct Node *head = NULL, *temp = NULL, *prev = NULL;

   printf("Enter number of nodes: \n");
   scanf("%d", &n);

   for(i = 0; i < n; i++) {
      temp = (struct Node*)malloc(sizeof(struct Node));
      printf("Enter data for node %d: \n", i+1);
      scanf("%d", &val);
      temp->data = val;
      temp->next = NULL;

      if(head == NULL){
         head = temp;
      }
      else{
         prev->next = temp;
      }
      prev = temp;
   }

   temp = head;
   printf("Result of Nodes : ");
   while(temp != NULL) {
      printf("%d -> ", temp->data);
      temp = temp->next;
   }
   printf("NULL\n");

   temp = head;
   while(temp != NULL) {
      struct Node* next = temp->next;
      free(temp);
      temp = next;
```

```
    }

    return 0;
}
```

Input & Output

```
Enter number of nodes:
3
Enter data for node 1:
34
Enter data for node 2:
20
Enter data for node 3:
12
Result of Nodes : 34 -> 20 -> 12 -> NULL
```

Q5) Write a C program to find the highest marks among n students using a structure array.

```
#include <stdio.h>

struct Student {
    char name[50];
    int marks;
};

int main() {
    int n, i, maxIndex = 0;
    printf("Enter number of students: ");
    scanf("%d", &n);

    struct Student students[n];

    for(i = 0; i < n; i++) {
        printf("\nEnter details of student %d\n", i+1);
        printf("Enter Name: ");
```

```
    scanf("%s", students[i].name);
    printf("Enter Marks: ");
    scanf("%d", &students[i].marks);
  }

  for(i = 1; i < n; i++) {
    if(students[i].marks > students[maxIndex].marks) {
      maxIndex = i;
    }
  }

  printf("\nStudent with highest marks:\n");
  printf("Name: %s\n", students[maxIndex].name);
  printf("Marks: %d\n", students[maxIndex].marks);

  return 0;
}
```

Input & Output

```
Enter number of students: 2


Enter details of student 1

Enter Name: Ohi

Enter Marks: 88


Enter details of student 2

Enter Name: Rav

Enter Marks: 87


Student with highest marks:

Name: Ohi

Marks: 8
```

Q6) How can you compare two structures in C? Illustrate with an example comparing two student structures.

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

struct Student {
   char name[50];
   int marks;
};

int main() {
   struct Student s1, s2;

   printf("Lets Compare Marks Between Two Students \n");
   printf("Enter name and marks of student 1 & student 2: ");
   scanf("%s %d", s1.name, &s1.marks);
   scanf("%s %d", s2.name, &s2.marks);

   if(strcmp(s1.name, s2.name) == 0 && s1.marks == s2.marks) {
      printf("The two students are equal.\n");
   } else {
      printf("The two students are not equal.\n");
   }

   return 0;
}
```

Input & Output

```
Lets Compare Marks Between Two Students
Enter name and marks of student 1 & student 2:
Ohi 92
Rav 88
The two students are not equal.
```

Q7) Write a program that uses an array of structures to represent a cricket team (player name, runs, wickets). Print the player with the highest runs.

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

struct Node {
   int data;
   struct Node* next;
};

int main() {
   int n, i, val;
   struct Node *head = NULL, *temp = NULL, *prev = NULL;

   printf("Enter number of nodes: \n");
   scanf("%d", &n);

   for(i = 0; i < n; i++) {
      temp = (struct Node*)malloc(sizeof(struct Node));
      printf("Enter data for node %d: \n", i+1);
      scanf("%d", &val);
      temp->data = val;
      temp->next = NULL;

      if(head == NULL){
         head = temp;
      }
      else{
         prev->next = temp;
      }
      prev = temp;
   }

   temp = head;
   printf("Result of Nodes : ");
   while(temp != NULL) {
      printf("%d -> ", temp->data);
      temp = temp->next;
   }
   printf("NULL\n");
```

```
    temp = head;
    while(temp != NULL) {
        struct Node* next = temp->next;
        free(temp);
        temp = next;
    }

    return 0;
}
```

Input & Output

```
Enter number of players: 2
Enter details of player 1
Name: Ohi
Runs: 220
Wickets: 34
Enter details of player 2
Name: Rav
Runs: 340
Wickets: 20
Player with highest runs:
Name: Rav
Runs: 340
Wickets: 20
```

Q8) Explain how calloc differs from malloc while allocating memory for an array of structures.

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

struct Student {
    char name[50];
    int marks;
};
```

```c
int main() {
   int n, i;
   printf("Enter number of students: ");
   scanf("%d", &n);

   struct Student* arr1 = (struct Student*) malloc(n * sizeof(struct Student));
   printf("\nUsing malloc:\n");
   for(i = 0; i < n; i++) {
      printf("Student %d initial marks: %d\n", i+1, arr1[i].marks);
   }

   struct Student* arr2 = (struct Student*) calloc(n, sizeof(struct Student));
   printf("\nUsing calloc:\n");
   for(i = 0; i < n; i++) {
      printf("Student %d initial marks: %d\n", i+1, arr2[i].marks);
   }

   free(arr1);
   free(arr2);

   return 0;
}
```

Input & Output

```
Enter number of students: 2

Using malloc:

Student 1 initial marks: 0

Student 2 initial marks: 0


Using calloc:

Student 1 initial marks: 0

Student 2 initial marks: 0
```

Q9) Can a struct contain a pointer to itself? Give an example.

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

struct Matrix {
   int rows;
   int cols;
   int **arr;
};

int main() {
   int i, j;
   struct Matrix m;

   printf("Enter number of rows and columns: ");
   scanf("%d %d", &m.rows, &m.cols);

   m.arr = (int**) malloc(m.rows * sizeof(int*));
   for(i = 0; i < m.rows; i++) {
      m.arr[i] = (int*) malloc(m.cols * sizeof(int));
   }

   printf("Enter elements of the matrix:\n");
   for(i = 0; i < m.rows; i++) {
      for(j = 0; j < m.cols; j++) {
         scanf("%d", &m.arr[i][j]);
      }
   }

   printf("\nMatrix elements:\n");
   for(i = 0; i < m.rows; i++) {
      for(j = 0; j < m.cols; j++) {
         printf("%d ", m.arr[i][j]);
      }
      printf("\n");
   }

   for(i = 0; i < m.rows; i++)
      free(m.arr[i]);
   free(m.arr);
```

```
    return 0;
}
```

Input & Output

```
Enter number of rows and columns:
3
2
Enter elements of the matrix:
6
12
34
20
2
14

Matrix elements:
6 12
34 20
2 14
```

Q10) Using realloc, write a program to increase the number of students in a dynamically allocated array of structures and input the new student details.

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

struct Student {
    char name[50];
    int marks;
};

int main() {
    int n, new_n, i;
    printf("Enter initial number of students: ");
```

```
   scanf("%d", &n);

   struct Student* students = (struct Student*) malloc(n * sizeof(struct
Student));

   for(i = 0; i < n; i++) {
      printf("\nEnter name and marks of student %d: ", i+1);
      scanf("%s %d", students[i].name, &students[i].marks);
   }

   printf("\nEnter number of new students to add: ");
   scanf("%d", &new_n);
   n += new_n;

   students = (struct Student*) realloc(students, n * sizeof(struct Student));

   for(i = n - new_n; i < n; i++) {
      printf("\nEnter name and marks of new student %d: ", i - (n - new_n) + 1);
      scanf("%s %d", students[i].name, &students[i].marks);
   }

   printf("\nAll students:\n");
   for(i = 0; i < n; i++) {
      printf("Student %d: %s, Marks: %d\n", i+1, students[i].name,
students[i].marks);
   }

   free(students);
   return 0;
}
```

Input & Output

```
Enter number of rows and columns:
3
2
Enter elements of the matrix:
6
12
34
```

```
20
2
14

Matrix elements:
6 12
34 20
2 14
```