Complex Number –

```c
//Implement Complex numbers using structures. Write unctions to add, multiply,
subtract two complex numbers.
#include <stdio.h>
#include <stdlib.h>
struct complex
{
  int real, img;
};

int main()
{
  int choice, x, y, z;
  struct complex a, b, c;

  while(1)
  {
    printf("Press 1 to add two complex numbers.\n");
    printf("Press 2 to subtract two complex numbers.\n");
    printf("Press 3 to multiply two complex numbers.\n");
    printf("Press 4 to divide two complex numbers.\n");
    printf("Press 5 to exit.\n");
    printf("Enter your choice\n");
    scanf("%d", &choice);

    if (choice == 5)
      exit(0);

    if (choice >= 1 && choice <= 4)
    {
      printf("Enter a and b where a + ib is the first complex number.");
      printf("\na = ");
      scanf("%d", &a.real);
      printf("b = ");
      scanf("%d", &a.img);
      printf("Enter c and d where c + id is the second complex number.");
      printf("\nc = ");
      scanf("%d", &b.real);
      printf("d = ");
      scanf("%d", &b.img);
    }
    if (choice == 1)
    {
      c.real = a.real + b.real;
      c.img = a.img + b.img;

      if (c.img >= 0)
        printf("Sum of the complex numbers = %d + %di", c.real, c.img);
      else
        printf("Sum of the complex numbers = %d %di", c.real, c.img);
```

```
  }
  else if (choice == 2)
  {
    c.real = a.real - b.real;
    c.img = a.img - b.img;

    if (c.img >= 0)
      printf("Difference of the complex numbers = %d + %di", c.real, c.img);
    else
      printf("Difference of the complex numbers = %d %di", c.real, c.img);
  }
  else if (choice == 3)
  {
    c.real = a.real*b.real - a.img*b.img;
    c.img = a.img*b.real + a.real*b.img;

    if (c.img >= 0)
      printf("Multiplication of the complex numbers = %d + %di", c.real, c.img);
    else
      printf("Multiplication of the complex numbers = %d %di", c.real, c.img);
  }
  else if (choice == 4)
  {
    if (b.real == 0 && b.img == 0)
      printf("Division by 0 + 0i isn't allowed.");
    else
    {
      x = a.real*b.real + a.img*b.img;
      y = a.img*b.real - a.real*b.img;
      z = b.real*b.real + b.img*b.img;

      if (x%z == 0 && y%z == 0)
      {
        if (y/z >= 0)
          printf("Division of the complex numbers = %d + %di", x/z, y/z);
        else
          printf("Division of the complex numbers = %d %di", x/z, y/z);
      }
      else if (x%z == 0 && y%z != 0)
      {
        if (y/z >= 0)
          printf("Division of two complex numbers = %d + %d/%di", x/z, y, z);
        else
          printf("Division of two complex numbers = %d %d/%di", x/z, y, z);
      }
      else if (x%z != 0 && y%z == 0)
      {
        if (y/z >= 0)
          printf("Division of two complex numbers = %d/%d + %di", x, z, y/z);
        else
          printf("Division of two complex numbers = %d %d/%di", x, z, y/z);
```

```
        }
        else
        {
          if (y/z >= 0)
            printf("Division of two complex numbers = %d/%d + %d/%di",x, z, y, z);
          else
            printf("Division of two complex numbers = %d/%d %d/%di", x, z, y, z);
        }
      }
    }
    else
      printf("Invalid choice.");

    printf("\nPress any key to enter choice again...\n");
  }
}
```

samuelDataEmployees

```
/* Samuel wants to store the data of his employees, which includes the following
fields:
(1) Name of the emplovee (11) Date of birth which is a collection of dav. month.
vear!
(in) Address which is a collection of house number. zip code and state!. Write a 'C
program to read and display the data of N emplovees using pointers to array of */

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

struct Date {
    int day;
    int month;
    int year;
};

struct Address {
    char houseNumber[20];
    char zipCode[20];
    char state[30];
};

struct Employee {
    char name[50];
    struct Date dob;
    struct Address address;
};

void readEmployee(struct Employee *employee) {
    printf("Enter employee name: ");
    scanf("%s", employee->name);
```

```
    printf("Enter date of birth (dd mm yyyy): ");
    scanf("%d %d %d", &employee->dob.day, &employee->dob.month, &employee-
>dob.year);

    printf("Enter house number: ");
    scanf("%s", employee->address.houseNumber);

    printf("Enter zip code: ");
    scanf("%s", employee->address.zipCode);

    printf("Enter state: ");
    scanf("%s", employee->address.state);
}

void displayEmployee(const struct Employee *employee) {
    printf("Name: %s\n", employee->name);
    printf("Date of Birth: %d/%d/%d\n", employee->dob.day, employee->dob.month,
employee->dob.year);
    printf("Address: %s, %s, %s\n", employee->address.houseNumber, employee-
>address.zipCode, employee->address.state);
}

int main() {
    int numEmployees;

    printf("Enter the number of employees: ");
    scanf("%d", &numEmployees);

    struct Employee *employees = (struct Employee *)malloc(numEmployees *
sizeof(struct Employee));
    if (employees == NULL) {
        printf("Memory allocation failed\n");
        return 1;
    }

    for (int i = 0; i < numEmployees; ++i) {
        printf("Enter details for employee %d:\n", i + 1);
        readEmployee(&employees[i]);
    }

    printf("\nEmployee details:\n");
    for (int i = 0; i < numEmployees; ++i) {
        printf("\nEmployee %d:\n", i + 1);
        displayEmployee(&employees[i]);
    }

    free(employees);

    return 0;
}
```

studentStruct

```c
//Write a C program to implement the following functions. Use pointers and dynamic
memory management functions
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct Student {
    char name[50];
    int rollNumber;
    float cgpa;
};

void readStudent(struct Student *student) {
    printf("Enter student name: ");
    scanf("%s", student->name);

    printf("Enter roll number: ");
    scanf("%d", &student->rollNumber);

    printf("Enter CGPA: ");
    scanf("%f", &student->cgpa);
}

void displayStudent(const struct Student *student) {
    printf("Name: %s\n", student->name);
    printf("Roll Number: %d\n", student->rollNumber);
    printf("CGPA: %.2f\n", student->cgpa);
}

int compareStudents(const void *a, const void *b) {
    const struct Student *studentA = (const struct Student *)a;
    const struct Student *studentB = (const struct Student *)b;

    return studentA->rollNumber - studentB->rollNumber;
}

int main() {
    int numStudents;

    printf("Enter the number of students: ");
    scanf("%d", &numStudents);

    struct Student *students = (struct Student *)malloc(numStudents * sizeof(struct
Student));
    if (students == NULL) {
        printf("Memory allocation failed\n");
```

```
        return 1;
    }

    for (int i = 0; i < numStudents; ++i) {
        printf("Enter details for student %d:\n", i + 1);
        readStudent(&students[i]);
    }

    printf("\nStudent details:\n");
    for (int i = 0; i < numStudents; ++i) {
        printf("\nStudent %d:\n", i + 1);
        displayStudent(&students[i]);
    }

    qsort(students, numStudents, sizeof(struct Student), compareStudents);

    printf("\nSorted by roll number:\n");
    for (int i = 0; i < numStudents; ++i) {
        printf("\nStudent %d:\n", i + 1);
        displayStudent(&students[i]);
    }

    free(students);

    return 0;
}
```

studentStruct

```
/* Create a structure STUDENT consisting of variables of structures:
i. DOB {day, month (use pointer), year). ii. STU_INFO (reg_no, name(use pointer),
address),
i. COLLEGE (college_name (use pointer), university_name )
where structure types from i to in are declared outside the STUDENT independently.
Show how to read and display member variables of DOB type if pointer variable is
created for
DOB inside STUDENT and STUDENT variable is also a pointer variable. The program
should read
and display the values of all members of STUDENT structure. */

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

// Structure for Date of Birth (DOB)
struct DOB {
    int day;
    int *month; // Using pointer for month
    int year;
};
```

```c
// Structure for Student Information (STU_INFO)
struct STU_INFO {
    int reg_no;
    char *name; // Using pointer for name
    char *address;
};

// Structure for College
struct COLLEGE {
    char *college_name;
    char *university_name;
};

// Structure for Student (containing nested structures)
struct STUDENT {
    struct DOB dob;
    struct STU_INFO stu_info;
    struct COLLEGE college;
};

int main() {
    // Declare a pointer to STUDENT structure
    struct STUDENT *student;

    // Allocate memory for student structure
    student = (struct STUDENT *)malloc(sizeof(struct STUDENT));
    if (student == NULL) {
        printf("Memory allocation failed\n");
        return 1;
    }

    // Allocate memory for nested structures
    student->dob.month = (int *)malloc(sizeof(int));
    student->stu_info.name = (char *)malloc(50 * sizeof(char));
    student->stu_info.address = (char *)malloc(100 * sizeof(char));
    student->college.college_name = (char *)malloc(50 * sizeof(char));
    student->college.university_name = (char *)malloc(50 * sizeof(char));

    // Read student details
    printf("Enter student's date of birth (day month year): ");
    scanf("%d %d %d", &student->dob.day, student->dob.month, &student->dob.year);

    printf("Enter student's registration number: ");
    scanf("%d", &student->stu_info.reg_no);

    printf("Enter student's name: ");
    scanf("%s", student->stu_info.name);

    printf("Enter student's address: ");
    scanf(" %[^\n]", student->stu_info.address);
```

```
    printf("Enter college name: ");
    scanf("%s", student->college.college_name);

    printf("Enter university name: ");
    scanf("%s", student->college.university_name);

    // Display student details
    printf("\nStudent details:\n");
    printf("Date of Birth: %d/%d/%d\n", student->dob.day, *(student->dob.month),
student->dob.year);
    printf("Registration Number: %d\n", student->stu_info.reg_no);
    printf("Name: %s\n", student->stu_info.name);
    printf("Address: %s\n", student->stu_info.address);
    printf("College: %s\n", student->college.college_name);
    printf("University: %s\n", student->college.university_name);

    // Free allocated memory
    free(student->dob.month);
    free(student->stu_info.name);
    free(student->stu_info.address);
    free(student->college.college_name);
    free(student->college.university_name);
    free(student);

    return 0;
}
```