ASSIGNMENT 4

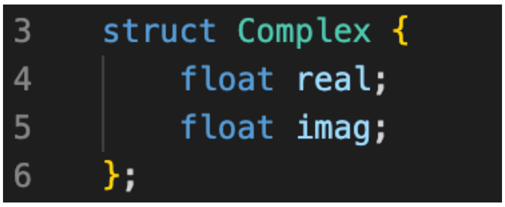
By – Dikshant

Roll no. - 2022102038

Question 1:

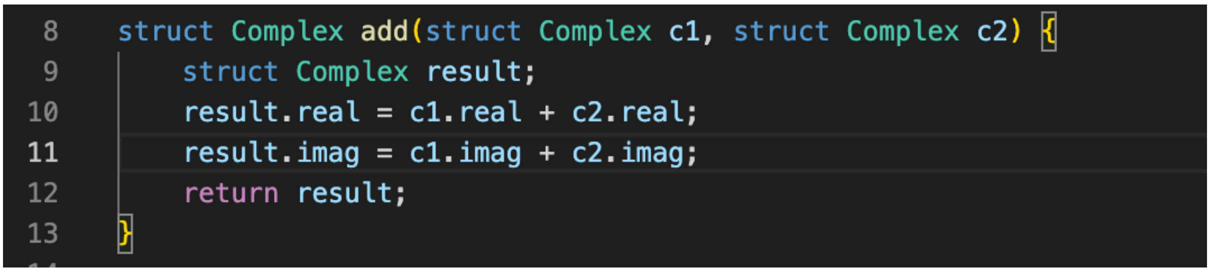
Aim: to perform addition, subtraction, multiplication of complex numbers.

Explanation: In the starting we have defined a struct “Complex” to represent a complex number. It has two members real (representing real part) and imag (imaginary part).



```
struct Complex {
    float real;
    float imag;
};
```

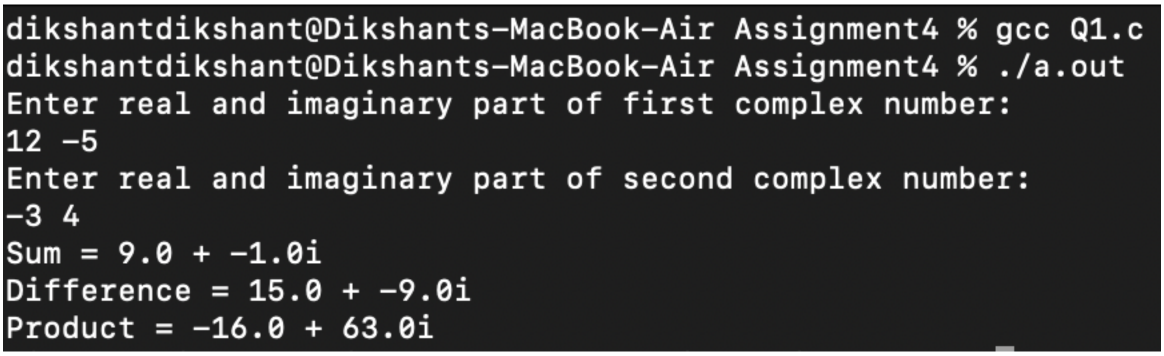
Then there is add() function which takes two complex structures as arguments and returns the result as a complex structure.



```
struct Complex add(struct Complex c1, struct Complex c2) {
    struct Complex result;
    result.real = c1.real + c2.real;
    result.imag = c1.imag + c2.imag;
    return result;
}
```

sub(), mul() functions also works as add() function.

The int main() function will ask the user to enter two complex numbers c1 and c2. And then it'll call out add(), sub() and mul() functions and display the result in x+iy form where x is the real part, y is imaginary part.

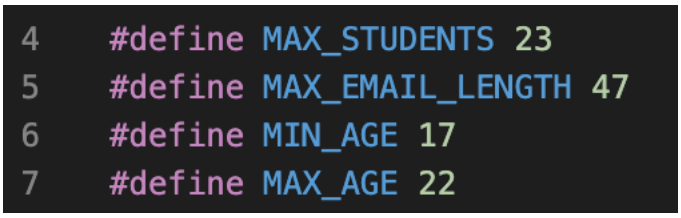


```
dikshantdikshant@Dikshants-MacBook-Air Assignment4 % gcc Q1.c
dikshantdikshant@Dikshants-MacBook-Air Assignment4 % ./a.out
Enter real and imaginary part of first complex number:
12 -5
Enter real and imaginary part of second complex number:
-3 4
Sum = 9.0 + -1.0i
Difference = 15.0 + -9.0i
Product = -16.0 + 63.0i
```

Question 2:

Aim: to store student info., printing names of students age 19 yrs., print all the details with email, printing names of all students who's email address starts with 'I','T'& have even numbers.

Explanation: We have defined maximum students entry that can be done in the database to be 23 (since hostel address of students was asked to be ≥20), age of students between 17 to 22 since it was also asked in the question.



```
#define MAX_STUDENTS 23
#define MAX_EMAIL_LENGTH 47
#define MIN_AGE 17
#define MAX_AGE 22
```

Then we have defined a struct “student”, it’ll store details of 23 students.

“add_student()” function ‘ll be called to add the student information to the “student” struct.
The function starts by copying the values of “name”, “ email”, “age”, “roll_no” and “hostel _address” into the corresponding fields of “student” struct at the “student_count” index. “student_count is a global variable that keeps tracks of the number of students added so far, and is incremented by 1 after each new student is added.

“print_students_with-age_19()” function will iterates over an array of “student” struct for each iteration it’ll check If the age is equal to 19. If it’s it’ll print the name of student. If not than it moves on and it ends when all the students have been checked.

Similar to the above “**print_students_with_email_IT_and_even()**” & “**display_student_details()**” works.

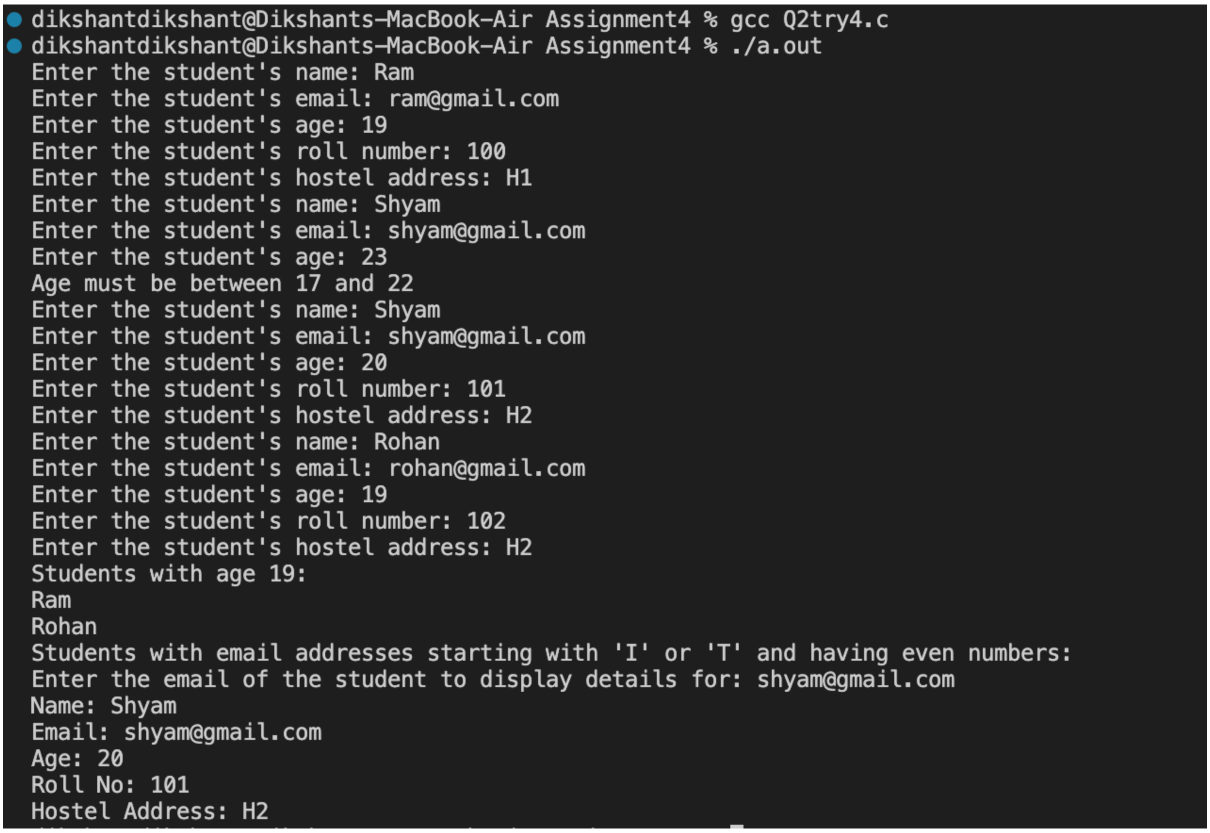
Int main() function repeatedly asks user to enter the information for each student until the number of students reach 23.

The “add_student” function ‘ll add the info to the array of students.

“print_students_with-age_19()” function is called to print the name of students with age 19. Similarly other 2 functions are also called.

The data of this isn’t stored in any file.

Sample output when max students were set to 3.



```
● dikshantdikshant@Dikshants-MacBook-Air Assignment4 % gcc Q2try4.c
● dikshantdikshant@Dikshants-MacBook-Air Assignment4 % ./a.out
  Enter the student's name: Ram
  Enter the student's email: ram@gmail.com
  Enter the student's age: 19
  Enter the student's roll number: 100
  Enter the student's hostel address: H1
  Enter the student's name: Shyam
  Enter the student's email: shyam@gmail.com
  Enter the student's age: 23
  Age must be between 17 and 22
  Enter the student's name: Shyam
  Enter the student's email: shyam@gmail.com
  Enter the student's age: 20
  Enter the student's roll number: 101
  Enter the student's hostel address: H2
  Enter the student's name: Rohan
  Enter the student's email: rohan@gmail.com
  Enter the student's age: 19
  Enter the student's roll number: 102
  Enter the student's hostel address: H2
  Students with age 19:
  Ram
  Rohan
  Students with email addresses starting with 'I' or 'T' and having even numbers:
  Enter the email of the student to display details for: shyam@gmail.com
  Name: Shyam
  Email: shyam@gmail.com
  Age: 20
  Roll No: 101
  Hostel Address: H2
```

Question 3:

Aim: to write a program to capitalize first and last letter of every word in a file.

Explanation:

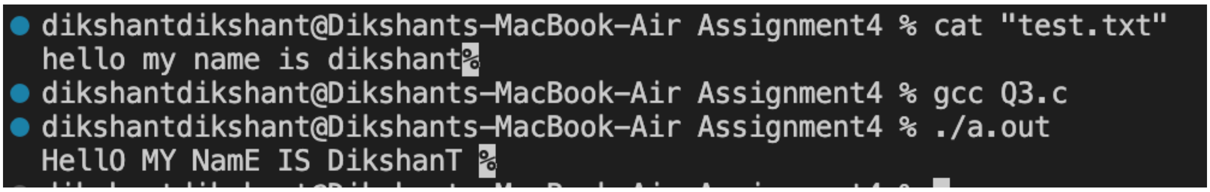
In the code we have included "ctype.h" header file because it contains a function "toupper" this function takes the integer representation of lowercase letters in ASCII and returns its uppercase equivalent.

In main function of this program we have declared a file pointer "fp", "fopen" is used to open the file "test.txt" in read mode, and the resulting file pointer is stores in "fp". If the file couldn't be opened then it'll return 1 to indicate error.

In "while loop" "fscan" function is used to read words from the file and store them in the "word" array this loop continues until it reaches EOF.

In the loop first & last letter of the word is capitalized using "toupper" function.

After all words have been processed the file is then closed using "fclose".

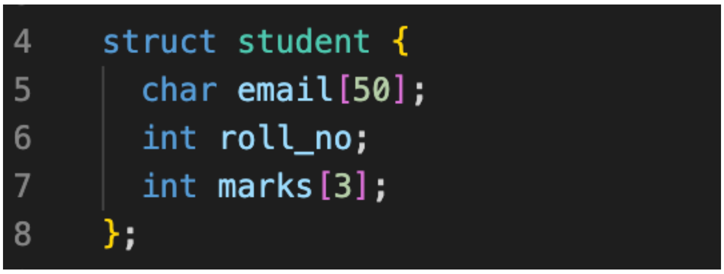


```
● dikshantdikshant@Dikshants-MacBook-Air Assignment4 % cat "test.txt"
  hello my name is dikshant%
● dikshantdikshant@Dikshants-MacBook-Air Assignment4 % gcc Q3.c
● dikshantdikshant@Dikshants-MacBook-Air Assignment4 % ./a.out
  HellO MY NamE IS DikshanT %
```

Question 4:

Aim: to write a program to read and write student email, roll no, marks of assignments in a file.

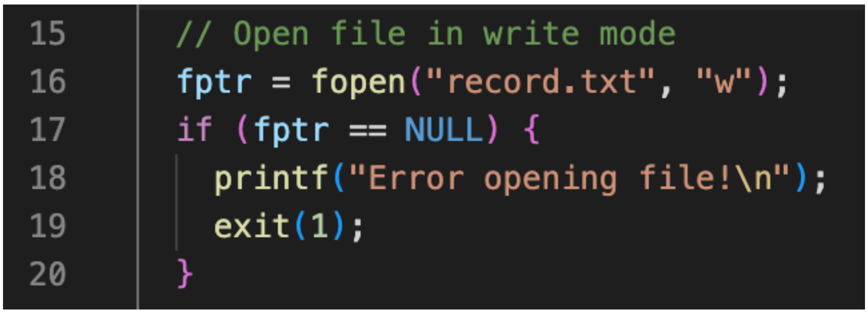
Explanation:



```
struct student {
    char email[50];
    int roll_no;
    int marks[3];
};
```

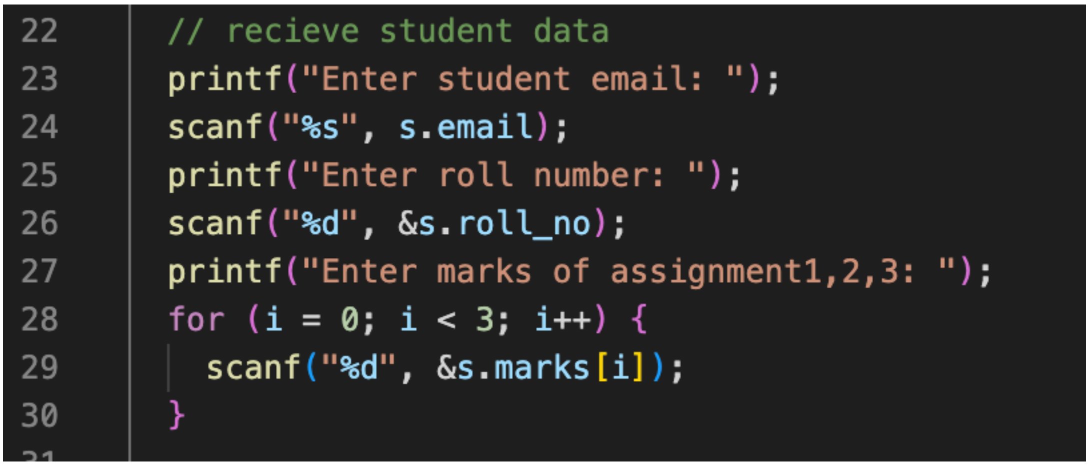
We have defined struct “students” that has email, roll_no, marks as its members. It will store the information as an array.
Struct student s will store the information of single student such as email, roll no, marks.

We have defined fptr as pointer to the file.
We have opened the file “record.txt” in write mode.



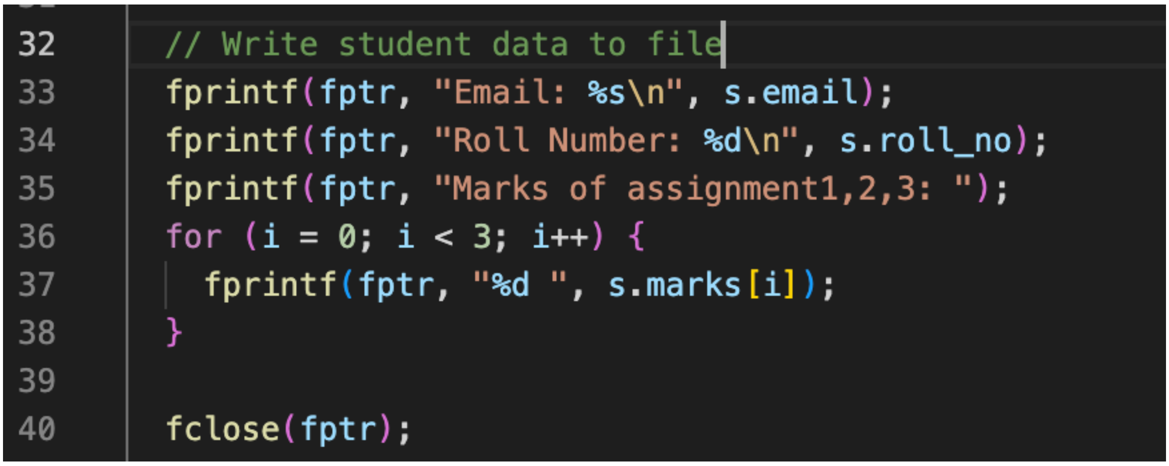
```
    // Open file in write mode
    fptr = fopen("record.txt", "w");
    if (fptr == NULL) {
        printf("Error opening file!\n");
        exit(1);
    }
```

The if loop will check if the file was successfully opened oe not. If the file pointer is equal to “Null”. It means that the file couldnot be opened and an error message is printed “exit(1)” function is then called which terminates the program to indicate error.



```
    // recieve student data
    printf("Enter student email: ");
    scanf("%s", s.email);
    printf("Enter roll number: ");
    scanf("%d", &s.roll_no);
    printf("Enter marks of assignment1,2,3: ");
    for (i = 0; i < 3; i++) {
        scanf("%d", &s.marks[i]);
    }
```

This allows the user to input data and store it in the struct “student”. For loop is used to read the values of the marks of assignment 1,2,3. It reads the integer value from user and store it in the “marks” (which is the member of struct “student”).

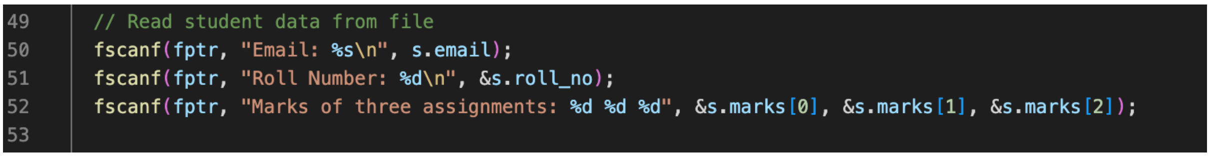


```
    // Write student data to file
    fprintf(fptr, "Email: %s\n", s.email);
    fprintf(fptr, "Roll Number: %d\n", s.roll_no);
    fprintf(fptr, "Marks of assignment1,2,3: ");
    for (i = 0; i < 3; i++) {
        fprintf(fptr, "%d ", s.marks[i]);
    }

    fclose(fptr);
```

This writes the student data to a file named “record.txt”. “f printf()” is used to write the data to the file, first argument is the file pointer that was opened in write mode and the following arguments are formatted strings and variables that are to be written to the file . after writing the data file is closed.

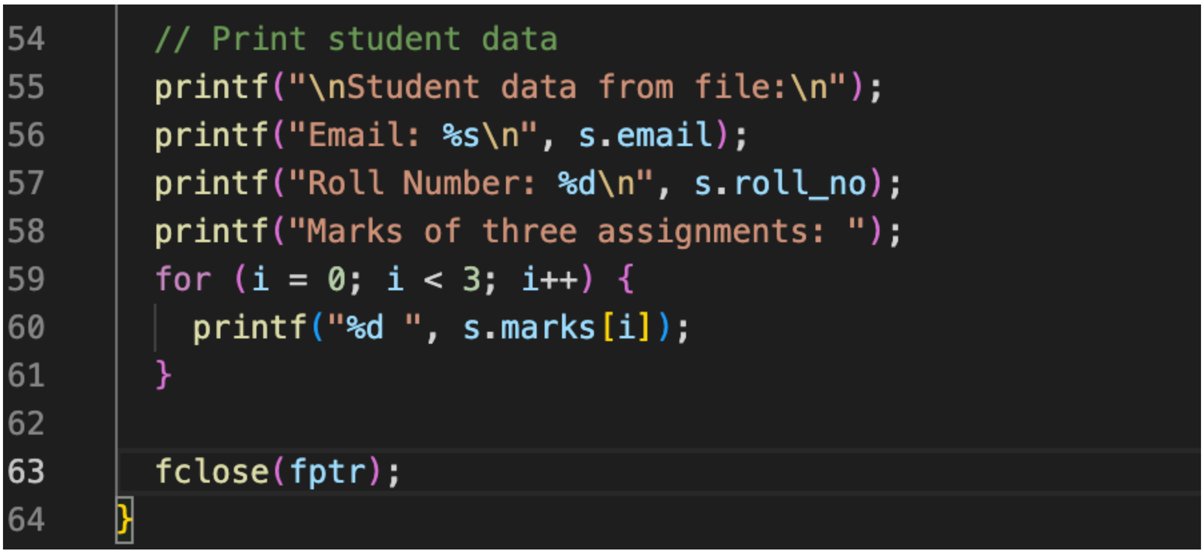
Then the file is again opened in read mode.



```c
    // Read student data from file
    fscanf(fptr, "Email: %s\n", s.email);
    fscanf(fptr, "Roll Number: %d\n", &s.roll_no);
    fscanf(fptr, "Marks of three assignments: %d %d %d", &s.marks[0], &s.marks[1], &s.marks[2]);

```

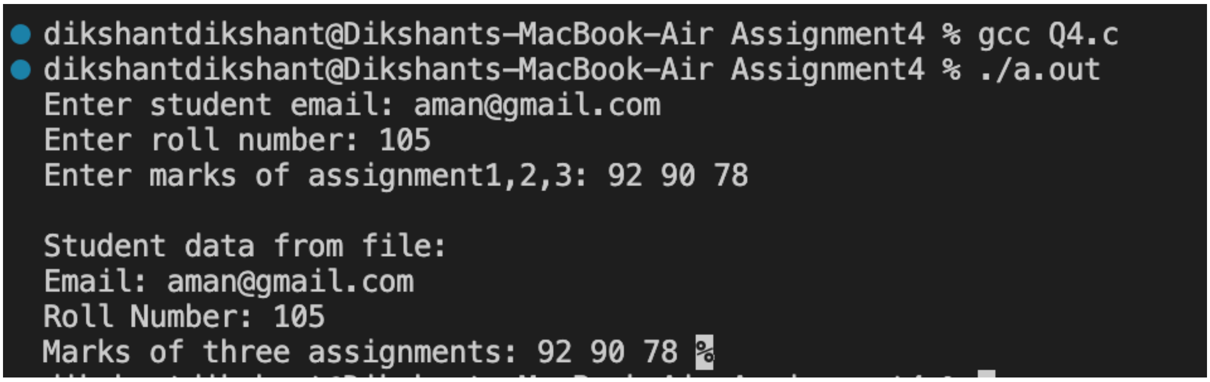
"fscanf()" will read the formatted input form the file.



```c
    // Print student data
    printf("\nStudent data from file:\n");
    printf("Email: %s\n", s.email);
    printf("Roll Number: %d\n", s.roll_no);
    printf("Marks of three assignments: ");
    for (i = 0; i < 3; i++) {
        printf("%d ", s.marks[i]);
    }

    fclose(fptr);
}
```

It will be printing student data that was stored in the file. It works in similar way 'write student data to file' was working.



```
● dikshantdikshant@Dikshants-MacBook-Air Assignment4 % gcc Q4.c
● dikshantdikshant@Dikshants-MacBook-Air Assignment4 % ./a.out
  Enter student email: aman@gmail.com
  Enter roll number: 105
  Enter marks of assignment1,2,3: 92 90 78

  Student data from file:
  Email: aman@gmail.com
  Roll Number: 105
  Marks of three assignments: 92 90 78 %
```