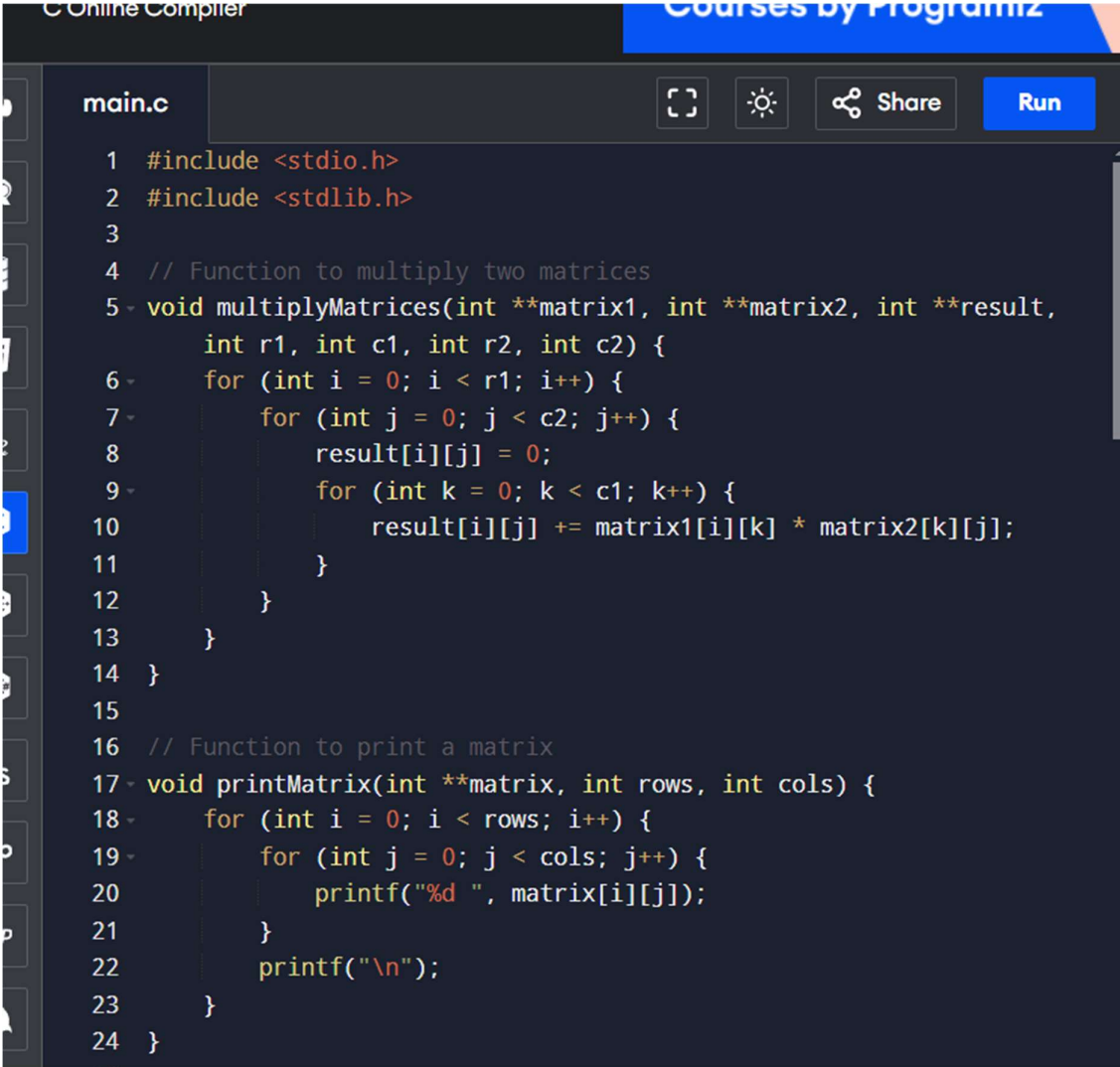


# DSA ASSIGNMENT 2

MANU KRISHNAN

RA2311026050065

## QUESTION 1:-



The screenshot shows an online C compiler interface. At the top, there's a header bar with "C Online Compiler" on the left and "Courses by Programiz" on the right. Below the header, there's a toolbar with icons for a code editor, a sun icon, a share icon, and a "Run" button. The main area displays a C program in a file named "main.c". The code defines two functions: "multiplyMatrices" and "printMatrix". The "multiplyMatrices" function takes two input matrices and a result matrix as pointers to arrays of integers, along with their dimensions. It uses three nested loops to calculate the product of the matrices. The "printMatrix" function takes a matrix pointer, its number of rows, and its number of columns, and prints the matrix elements in a formatted way, with spaces between elements and a newline at the end of each row. The code is as follows:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 // Function to multiply two matrices
5 void multiplyMatrices(int **matrix1, int **matrix2, int **result,
6     int r1, int c1, int r2, int c2) {
7     for (int i = 0; i < r1; i++) {
8         for (int j = 0; j < c2; j++) {
9             result[i][j] = 0;
10            for (int k = 0; k < c1; k++) {
11                result[i][j] += matrix1[i][k] * matrix2[k][j];
12            }
13        }
14    }
15
16 // Function to print a matrix
17 void printMatrix(int **matrix, int rows, int cols) {
18     for (int i = 0; i < rows; i++) {
19         for (int j = 0; j < cols; j++) {
20             printf("%d ", matrix[i][j]);
21         }
22         printf("\n");
23     }
24 }
```

main.c



Share

Run

```
25
26 int main() {
27     int r1, c1, r2, c2;
28
29     // Get the dimensions of the first matrix
30     printf("Enter the number of rows for the first matrix: ");
31     scanf("%d", &r1);
32     printf("Enter the number of columns for the first matrix: ");
33     scanf("%d", &c1);
34
35     // Get the dimensions of the second matrix
36     printf("Enter the number of rows for the second matrix: ");
37     scanf("%d", &r2);
38     printf("Enter the number of columns for the second matrix: ");
39     scanf("%d", &c2);
40
41     // Check if the matrices can be multiplied
42     if (c1 != r2) {
43         printf("The matrices cannot be multiplied.\n");
44         return 1;
45     }
46
47     // Dynamically allocate memory for the matrices
48     int **matrix1 = (int **)malloc(r1 * sizeof(int *));
49     for (int i = 0; i < r1; i++) {
50         matrix1[i] = (int *)malloc(c1 * sizeof(int));
```

main.c

File

Edit

View

Run

```
49- for (int i = 0; i < r1; i++) {
50-     matrix1[i] = (int *)malloc(c1 * sizeof(int));
51- }
52-
53- int **matrix2 = (int **)malloc(r2 * sizeof(int *));
54- for (int i = 0; i < r2; i++) {
55-     matrix2[i] = (int *)malloc(c2 * sizeof(int));
56- }
57-
58- int **result = (int **)malloc(r1 * sizeof(int *));
59- for (int i = 0; i < r1; i++) {
60-     result[i] = (int *)malloc(c2 * sizeof(int));
61- }
62-
63- // Get the elements of the first matrix
64- printf("Enter the elements of the first matrix:\n");
65- for (int i = 0; i < r1; i++) {
66-     for (int j = 0; j < c1; j++) {
67-         scanf("%d", &matrix1[i][j]);
68-     }
69- }
70-
71- // Get the elements of the second matrix
72- printf("Enter the elements of the second matrix:\n");
73- for (int i = 0; i < r2; i++) {
74-     for (int j = 0; j < c2; j++) {
```

```
main.c
74     for (int j = 0; j < c2; j++) {
75         scanf("%d", &matrix2[i][j]);
76     }
77 }
78
79 // Multiply the matrices
80 multiplyMatrices(matrix1, matrix2, result, r1, c1, r2, c2);
81
82 // Print the result
83 printf("The result of the matrix multiplication is:\n");
84 printMatrix(result, r1, c2);
85
86 // Free the dynamically allocated memory
87 for (int i = 0; i < r1; i++) {
88     free(matrix1[i]);
89 }
90 free(matrix1);
91
92 for (int i = 0; i < r2; i++) {
93     free(matrix2[i]);
94 }
95 free(matrix2);
96
97 for (int i = 0; i < r1; i++) {
98     free(result[i]);
99 }
100 free(result);
101
102 return 0;
103 }
```

SAMPLE OUTPUT:

```
^ /tmp/PdQXOpPywj.o
Enter the number of rows for the first matrix: 2
Enter the number of columns for the first matrix: 2
Enter the number of rows for the second matrix: 2
Enter the number of columns for the second matrix: 2
Enter the elements of the first matrix:
1
2
3
4
Enter the elements of the second matrix:
5
6
7
8
The result of the matrix multiplication is:
19 22
43 50

=== Code Execution Successful ===
```

QUESTION 2:-

main.c



Share

Run

```
1 #include <stdio.h>
2 #include <string.h>
3
4 #define MAX_STUDENTS 100
5 #define MAX_NAME_LENGTH 50
6
7 char students[MAX_STUDENTS][MAX_NAME_LENGTH];
8 int num_students = 0;
9
10 void create_list() {
11     printf("Enter the number of students: ");
12     scanf("%d", &num_students);
13     for (int i = 0; i < num_students; i++) {
14         printf("Enter student %d name: ", i + 1);
15         scanf("%s", students[i]);
16     }
17     printf("List created successfully!\n");
18 }
19
20 void insert_student() {
21     char name[MAX_NAME_LENGTH];
22     int index;
23     printf("Enter the new student's name: ");
24     scanf("%s", name);
25     printf("Enter the index to insert the student: ");
```

```

25     printf("Enter the index to insert the student: ");
26     scanf("%d", &index);
27     if (index < 0 || index > num_students) {
28         printf("Invalid index. Please try again.\n");
29     } else {
30         for (int i = num_students; i > index; i--) {
31             strcpy(students[i], students[i - 1]);
32         }
33         strcpy(students[index], name);
34         num_students++;
35         printf("Student inserted successfully!\n");
36     }
37 }
38
39 void delete_student() {
40     char choice;
41     printf("Do you want to delete by name or by index? (name/index
42         ): ");
43     scanf(" %c", &choice);
44     if (choice == 'n' || choice == 'N') {
45         char name[MAX_NAME_LENGTH];
46         printf("Enter the student's name to delete: ");
47         scanf("%s", name);
48         for (int i = 0; i < num_students; i++) {
49             if (strcmp(students[i], name) == 0) {
50                 for (int j = i; j < num_students - 1; j++) {

```



```
49     for (int j = 1; j < num_students - 1; j++) {
50         strcpy(students[j], students[j + 1]);
51     }
52     num_students--;
53     printf("Student deleted successfully!\n");
54     return;
55 }
56 }
57 printf("Student not found.\n");
58 } else if (choice == 'i' || choice == 'I') {
59     int index;
60     printf("Enter the index to delete the student: ");
61     scanf("%d", &index);
62     if (index < 0 || index >= num_students) {
63         printf("Invalid index. Please try again.\n");
64     } else {
65         for (int i = index; i < num_students - 1; i++) {
66             strcpy(students[i], students[i + 1]);
67         }
68         num_students--;
69         printf("Student deleted successfully!\n");
70     }
71 } else {
72     printf("Invalid choice. Please try again.\n");
73 }
74 }
```

```

74 }
75
76 void traverse_list() {
77     printf("Current list of students:\n");
78     for (int i = 0; i < num_students; i++) {
79         printf("%d. %s\n", i + 1, students[i]);
80     }
81 }
82
83 void search_student() {
84     char name[MAX_NAME_LENGTH];
85     printf("Enter the student's name to search: ");
86     scanf("%s", name);
87     for (int i = 0; i < num_students; i++) {
88         if (strcmp(students[i], name) == 0) {
89             printf("Student found at position %d.\n", i + 1);
90             return;
91         }
92     }
93     printf("Student not found.\n");
94 }
95
96 int main() {
97     while (1) {
98         printf("\nStudent Management System Menu:\n");
99         printf("1. Create list\n");

```

```

98     printf("\nStudent Management System Menu:\n");
99     printf("1. Create list\n");
100    printf("2. Insert student\n");
101    printf("3. Delete student\n");
102    printf("4. Traverse list\n");
103    printf("5. Search student\n");
104    printf("6. Exit\n");
105    int choice;
106    printf("Enter your choice: ");
107    scanf("%d", &choice);
108    switch (choice) {
109        case 1:
110            create_list();
111            break;
112        case 2:
113            insert_student();
114            break;
115        case 3:
116            delete_student();
117            break;
118        case 4:
119            traverse_list();
120            break;
121        case 5:
122            search_student();
123            break;
124        case 6:
125            break;
126        default:
127            printf("Invalid choice. Please try again.\n");
128    }
129    traverse_list();
130 }
131 return 0;
132 }

```

SAMPLE OUTPUT:-

/tmp/05Zd0UJQUc.o

Student Management System Menu:

1. Create list
2. Insert student
3. Delete student
4. Traverse list
5. Search student
6. Exit

Enter your choice: 1

Enter the number of students: 3

Enter student 1 name: john

Enter student 2 name: alice

Enter student 3 name: bob

List created successfully!

Current list of students:

1. john
2. alice
3. bob

Student Management System Menu:

1. Create list
2. Insert student
3. Delete student
4. Traverse list
5. Search student

```
4. Traverse list
5. Search student
6. Exit
Enter your choice: 2
Enter the new student's name: mike
Enter the index to insert the student: 1
Student inserted successfully!
Current list of students:
1. john
2. mike
3. alice
4. bob

Student Management System Menu:
1. Create list
2. Insert student
3. Delete student
4. Traverse list
5. Search student
6. Exit
Enter your choice: 3
Do you want to delete by name or by index? (name/index): name
Enter the student's name to delete: Student not found.
Current list of students:
1. john
2. mike
3. alice
```

- 2. mike
- 3. alice
- 4. bob

Student Management System Menu:

- 1. Create list
- 2. Insert student
- 3. Delete student
- 4. Traverse list
- 5. Search student
- 6. Exit

Enter your choice: 4

Current list of students:

- 1. john
- 2. mike
- 3. alice
- 4. bob

Current list of students:

- 1. john
- 2. mike
- 3. alice
- 4. bob

Student Management System Menu:

- 1. Create list
- 2. Insert student

```
1. Create list
2. Insert student
3. Delete student
4. Traverse list
5. Search student
6. Exit
Enter your choice: 5
Enter the student's name to search: alice
Student found at position 3.
Current list of students:
1. john
2. mike
3. alice
4. bob

Student Management System Menu:
1. Create list
2. Insert student
3. Delete student
4. Traverse list
5. Search student
6. Exit
Enter your choice: 6

=== Code Execution Successful ===
```