# ASSIGNMEN-2

**Name:Ashwin.B**

**Reg no:RA2311003050382**

**Year & Branch: II B.Tech CSE - F Section**

**Assignment No: 2**

**ASSIGNMENT-2**

**Implementation of Matrix Multiplication using Dynamic Memory Allocation. Ensureto allocate the memory using appropriate functions and access the array using pointers.**

**ANS:**

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

// Function to dynamically allocate memory for a matrix
Int** allocate_matrix(int rows, int cols) {
   Int **matrix = (int**)malloc(rows * sizeof(int*));
   For(int I = 0; I < rows; i++) {
      Matrix[i] = (int*)malloc(cols * sizeof(int));
   }
   Return matrix;
}
```

```
// Function to free allocated memory for a matrix
Void free_matrix(int **matrix, int rows) {
   For(int I = 0; I < rows; i++) {
      Free(matrix[i]);
   }
   Free(matrix);
}

// Function to input values into the matrix
Void input_matrix(int **matrix, int rows, int cols, const char *name) {
   Printf("Enter values for matrix %s:\n", name);
   For(int I = 0; I < rows; i++) {
      For(int j = 0; j < cols; j++) {
         Scanf("%d", &matrix[i][j]);
      }
   }
}

// Function to print the matrix
Void print_matrix(int **matrix, int rows, int cols) {
   For(int I = 0; I < rows; i++) {
```

```
      For(int j = 0; j < cols; j++) {
         Printf("%d\t", matrix[i][j]);
      }
      Printf("\n");
   }
}

// Function to multiply two matrices
Int** multiply_matrices(int **a, int **b, int rows, int cols) {
   Int **result = allocate_matrix(rows, cols);
   For(int I = 0; I < rows; i++) {
      For(int j = 0; j < cols; j++) {
         Result[i][j] = 0;
         For(int k = 0; k < cols; k++) {
            Result[i][j] += a[i][k] * b[k][j];
         }
      }
   }
   Return result;
}

    Int main() {
   Int rows, cols;
```

```
// Input number of rows and columns
Printf(“Enter the number of rows: “);
Scanf(“%d”, &rows);
Printf(“Enter the number of columns: “);
Scanf(“%d”, &cols);

// Allocate memory for matrices
Int **a = allocate_matrix(rows, cols);
Int **b = allocate_matrix(rows, cols);

// Input matrix values
Input_matrix(a, rows, cols, “A”);
Input_matrix(b, rows, cols, “B”);

// Perform matrix multiplication
Printf(“Matrix multiplication result:\n”);
Int **result = multiply_matrices(a, b, rows, cols);
Print_matrix(result, rows, cols);

// Free allocated memory
Free_matrix(a, rows);
Free_matrix(b, rows);
```

**Free_matrix(result, rows);**

**Return 0;**

**}**

**OUTPUT:**

```
Enter the number of rows:22
Enter the number of columns: 2
Enter values for matrix A:
2 2

1 1
Enter values for matrix B:
1 1

2 2
Matrix multiplication result:
3   4
2   3


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

**2. You are given a task with creating a simple student management system using arraysthat will allow the user to**

**manage student names. Implement the following operationson a list of student names using switch-case and arrays. After every operation,display the current list of students.**

**The operations to implement are:**

**(i) Creation of the list: Allow the user to create a list of student names by enteringthem one by one.**

**(ii) Insertion of a new student: Insert a new student's name into a specific positionin the list. The user should provide the name and the index at which it shouldbe inserted.**

**(iii) Deletion of a student: Delete a student's name from the list based on theirposition or name. Ask the user whether they want to delete by name or by index.**

**(iv) Traversal of the list: Display all the student names in the current order.**

**(v) Search for a student: Search for a student's name in the list and displaywhether or not the student is found, along with their position if present.**

**ANS:**

```
#include <stdio.h>

#include <string.h>

#define MAX 100

#define MAX_NAME_LEN 50


Void displayList(char students[][MAX_NAME_LEN], int size) {

  Printf(“Student list: “);
```

```
    For (int I = 0; I < size; i++) {
        Printf(“%s”, students[i]);
        If (I < size – 1) printf(“, “);
    }
    Printf(“\n”);
}

Void createList(char students[][MAX_NAME_LEN], int *size) {
    Printf(“Enter the number of students: “);
    Scanf(“%d”, size);
    For (int I = 0; I < *size; i++) {
        Printf(“Enter name of student %d: “, I + 1);
        Scanf(“%s”, students[i]);
    }
}

Void insertStudent(char students[][MAX_NAME_LEN], int
*size) {
    Char name[MAX_NAME_LEN];
    Int pos;
    Printf(“Enter name to insert: “);
    Scanf(“%s”, name);
    Printf(“Enter position (0-based index): “);
```

```
    Scanf(“%d”, &pos);

    If (pos < 0 || pos > *size) {
        Printf(“Invalid position!\n”);
        Return;
    }

    For (int I = *size; I > pos; i--) {
        Strcpy(students[i], students[I – 1]);
    }
    Strcpy(students[pos], name);
    (*size)++;
}

Void deleteStudent(char students[][MAX_NAME_LEN], int
*size) {
    Int pos;
    Printf(“Enter position (0-based index) to delete: “);
    Scanf(“%d”, &pos);

    If (pos < 0 || pos >= *size) {
        Printf(“Invalid position!\n”);
        Return;
```

```
    }

    For (int I = pos; I < *size – 1; i++) {
        Strcpy(students[i], students[I + 1]);
    }
    (*size)--;
}

Void searchStudent(char students[][MAX_NAME_LEN], int
size) {
    Char name[MAX_NAME_LEN];
    Printf(“Enter name to search: “);
    Scanf(“%s”, name);

    For (int I = 0; I < size; i++) {
        If (strcmp(students[i], name) == 0) {
            Printf(“%s found at position %d\n”, name, i);
            Return;
        }
    }
    Printf(“%s not found!\n”, name);
}
```

```
Int main() {
   Char students[MAX][MAX_NAME_LEN];
   Int size = 0, choice;

   Do {
      Printf(“\n1. Create list\n2. Insert student\n3. Delete student\n4. Display list\n5. Search student\n6. Exit\n”);
      Printf(“Enter your choice: “);
      Scanf(“%d”, &choice);

      Switch (choice) {
         Case 1: createList(students, &size); break;
         Case 2: insertStudent(students, &size); break;
         Case 3: deleteStudent(students, &size); break;
         Case 4: displayList(students, size); break;
         Case 5: searchStudent(students, size); break;
         Case 6: printf(“Exiting...\n”); break;
         Default: printf(“Invalid choice!\n”);
      }
   } while (choice != 6);

   Return 0;
}
```

**OUTPUT:**

```
1. Create list
2. Insert student
3. Delete student
4. Display list
5. Search student
6. Exit
Enter your choice: 1
Enter the number of students: 3
Enter name of student 1: Ashwin
Enter name of student 2: rahul
Enter name of student 3: avi

1. Create list
2. Insert student
3. Delete student
4. Display list
5. Search student
6. Exit
Enter your choice:
```