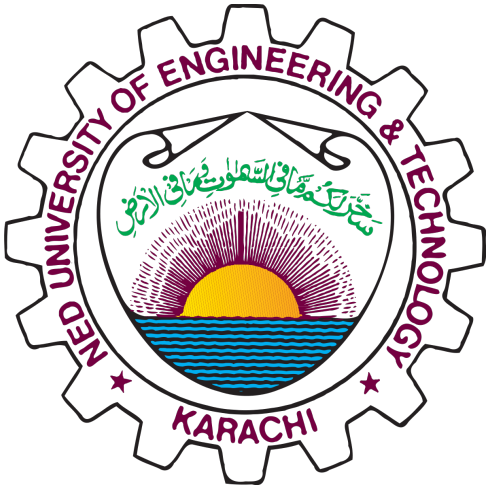

NED University of Engineering and Technology

Data Structures and Algorithms - Lab 1

Author: NAME
Roll No: CT-24000
Date: November 01, 2025

Department of Computer Science and Information Technology
Bachelor of Science (BS)

Question 1

```
#include <iostream>
using namespace std;

class Array2D {
    private:
        int **arr;
        int rows, cols;
    public:
        Array2D(int r, int c) : rows(r), cols(c) {
            arr = new int*[rows];
            for (int i = 0; i < rows; i++) {
                arr[i] = new int[cols];
            }
        }
        int* operator[](int index) {
            return arr[index];
        }
        int getRows() { return rows; }
        int getCols() { return cols; }
        ~Array2D() {
            for (int i = 0; i < rows; i++) {
                delete[] arr[i];
            }
            delete[] arr;
        }
};

class Array1D {
    private:
        int *arr;
        int size;
    public:
        Array1D(int s) : size(s) {
            arr = new int[size];
        }
        // Copy constructor that takes Array2D and flattens it (column order)
        Array1D(Array2D &other) {
            size = other.getRows() * other.getCols();
            arr = new int[size];
            int index = 0;
            for (int i = 0; i < other.getRows(); i++) {
                for (int j = 0; j < other.getCols(); j++) {
                    arr[index++] = other[j][i];
                }
            }
        }
        int& operator[](int index) {
            return arr[index];
        }
        ~Array1D() {
            delete[] arr;
        }
};

int main() {
```

```
    cout << "Enter number of rows and columns for 2D array: ";
    int r, c;
    cin >> r >> c;
    Array2D arr2d(r, c);
    for (int i = 0; i < r; i++) {
        for (int j = 0; j < c; j++) {
            cout << "Enter element at arr[" << i << "][" << j << "]: ";
            cin >> arr2d[i][j];
        }
    }
    Array1D arr1d = arr2d;
    cout << "Flattened 1D array elements: \n";
    for (int i = 0; i < r * c; i++) {
        cout << arr1d[i] << " ";
    }
}
```

Output

```
Enter number of rows and columns for 2D array:
Enter element at arr[0][0]:
Enter element at arr[0][1]:
Enter element at arr[1][0]:
Enter element at arr[1][1]:
Flattened 1D array elements:

1 2 3 4

[INPUT(S) PROVIDED]
2 2
1 3 2 4
```

Question 2

```
#include <iostream>
using namespace std;

class Table {
    private:
        float **table;
        int rows, cols, curr_row;

    public:
        Table(int r, int c) : rows(r), cols(c), curr_row(0) {
            table = new float*[rows];
            for (int i=0; i<cols; i++){
                table[i] = new float[cols];
            }
        }
        bool insert(float gpas[], int n){
            for (int i=0; i<n; i++){
                table[curr_row][i] = gpas[i];
            }
```

```
            curr_row++;
            return true;
        }
        void get_sem_gpas() {
            for (int i=0; i<rows; i++){
                float gpa=0;
                for (int j=0; j<cols; j++){
                    gpa += table[i][j];
                }
                cout << "GPA for student " << i+1 << " is " << gpa/cols << endl;
            }
        }
};

int main(){
    int rows, cols;
    cout << "Enter number of students: "; cin >> rows;
    cout << "Enter number of subjects: "; cin >> cols;
    Table students(rows, cols);

    for (int i = 0; i < rows; i++) {
        float gpas[3];
        cout << "Enter " << cols << " GPAs for student " << (i+1) << ": ";
        for (int j = 0; j < cols; j++) {
            cin >> gpas[j];
        }
        students.insert(gpas, cols);
    }

    students.get_sem_gpas();
    return 0;
}
```

Output

```
Enter number of students:
Enter number of subjects:
Enter 3 GPAs for student 1:
Enter 3 GPAs for student 2:
GPA for student 1 is 3.63333
GPA for student 2 is 3.16667

[INPUT(S) PROVIDED]
2 3
3.2 3.7 4
2.3 4 3.2
```

Question 3

```
#include <iostream>
#include <algorithm>
using namespace std;
class MedianFinder {
```

```
    private:
        int* arr;
        int size;
        int capacity;
    public:
        MedianFinder() {
            capacity = 100;
            arr = new int[capacity];
            size = 0;
        }
        void addNum(int num) {
            if (size == capacity) {
                capacity *= 2;
                int* newArr = new int[capacity];
                for (int i = 0; i < size; i++) {
                    newArr[i] = arr[i];
                }
                delete[] arr;
                arr = newArr;
            }
            arr[size++] = num;
        }
        double findMedian() {
            sort(arr, arr + size);
            if (size % 2 == 0) {
                return (arr[size / 2 - 1] + arr[size / 2]) / 2.0;
            } else {
                return arr[size / 2];
            }
        }
        ~MedianFinder() {
            delete[] arr;
        }


};

int main() {
    MedianFinder* medianFinder = new MedianFinder();

    int choice;
    while (true) {
        cout << "1) addNum\n2) findMedian\n3) quit\n=> "; cin >> choice;

        switch (choice) {
            case 1:
                int num;
                cout << "Enter number: "; cin >> num;
                medianFinder->addNum(num);
                cout << endl;
                break;

            case 2:
                cout << "The median is: " << medianFinder->findMedian() << endl << endl;
                break;

            default:
```

```
                delete medianFinder;
                return 0;
            }

        }

        delete medianFinder;
        return 0;
}
```

Output

```
1) addNum
2) findMedian
3) quit
=> Enter number:

1) addNum
2) findMedian
3) quit
=> Enter number:

1) addNum
2) findMedian
3) quit
=> Enter number:

1) addNum
2) findMedian
3) quit
=> The median is:
5

1) addNum
2) findMedian
3) quit
=>

[INPUT(S) PROVIDED]
1
5
1
2
1
5
2
3
```

Question 4

```
#include <iostream>
using namespace std;
class Array1D {
```

```
    private:
        int *arr;
        int size;
    public:
        Array1D(int s) : size(s) {
            arr = new int[size];
        }
        int& operator[](int index) {
            return arr[index];
        }
        // Binary search
        int find(int key) {
            int l=0, r=size-1;
            while (l <= r) {
                int mid = l + (r - l) / 2;
                if (arr[mid] == key) return mid;
                else if (arr[mid] < key) l = mid + 1;
                else r = mid - 1;
            }
            return -1;
        }
        ~Array1D() {
            delete[] arr;
        }
};

int main() {
    cout << "Enter size of 1D array: ";
    int s;
    cin >> s;
    Array1D arr(s);
    cout << "Enter elements in sorted order: ";
    for (int i = 0; i < s; i++) {
        cin >> arr[i];
    }
    cout << "Enter element to search: ";
    int key;
    cin >> key;
    int index = arr.find(key);
    if (index != -1) {
        cout << "Element found at index " << index << endl;
    } else {
        cout << "Element not found" << endl;
    }
    return 0;
}
```

Output

```
Enter size of 1D array:
Enter elements in sorted order:
Enter element to search:
Element found at index 2

[INPUT(S) PROVIDED]
5
2 13 16 23 35
16
```

Question 5

```
#include <iostream>
using namespace std;

class Array2D {
    private:
        int **arr;
        int rows, cols;
    public:
        Array2D(int r, int c) : rows(r), cols(c) {
            arr = new int*[rows];
            for (int i = 0; i < rows; i++) {
                arr[i] = new int[cols];
            }
        }
        int* operator[](int index) {
            return arr[index];
        }
        int find(int key) {
            int l = 0, r = rows * cols - 1;
            while (l <= r) {
                int mid = l + (r - l) / 2;
                int mid_val = arr[mid / cols][mid % cols];
                if (mid_val == key) return true;
                else if (mid_val < key) l = mid + 1;
                else r = mid - 1;
            }
            return false;
        }
        ~Array2D() {
            for (int i = 0; i < rows; i++) {
                delete[] arr[i];
            }
            delete[] arr;
        }
};

int main() {
    cout << "Enter number of rows and columns for 2D array: ";
    int r, c;
    cin >> r >> c;
    Array2D arr2d(r, c);
```

```
    cout << "Enter elements in sorted order:\n";
    for (int i = 0; i < r; i++) {
        for (int j = 0; j < c; j++) {
            cin >> arr2d[i][j];
        }
    }
    cout << "Enter element to search: ";
    int key;
    cin >> key;
    int index = arr2d.find(key);
    if (index) {
        cout << "Element found" << endl;
    } else {
        cout << "Element not found" << endl;
    }
    return 0;
}
```

Output

```
Enter number of rows and columns for 2D array:
Enter elements in sorted order:
Enter element to search:
Element found

[INPUT(S) PROVIDED]
2 2
2 5 10 15
5
```