Assignment - 1

Name: AKSHAT JAIMINI

Roll Number: 102103586

Q1.

```
#include <iostream>

using namespace std;

int A[100];

void create(int len){
    cout << "Enter elemet at: " << endl;
    for(int i = 0; i  < len; i++){
        cout << i << ": ";
        cin >> A[i];
    }
}

void display(int len){
    for(int i = 0; i < len; i++){
        cout << A[i] << ", ";
    }
    //cout << endl;
}

int insert(int elem, int pos, int len){
    for(int i = len-1; i >= pos; i--){
        A[i+1] = A[i];
    }
    A[pos] = elem;
    return len+1;
```

```
}

int delete_elem(int pos, int len){
    for(int i = pos; i < len; i++){
        A[i] = A[i+1];
    }
    return len-1;
}

int linear_search(int len, int elem){
    for(int i = 0; i < len; i++){
        if(A[i] == elem){
            return i;
        }
    }
    return -1;
}

int binary_search(int len, int elem){
    int low = 0;
    int high = len-1;
    while(low <= high){
        int mid = (low+high)/2;
        if(A[mid] == elem){
            return mid;
        }else if(elem > A[mid]){
            low = mid + 1;
        }else if(elem < A[mid]){
            high = mid-1;
        }
    }
    return -1;
}

int main(){
    int len = 10;
    bool terminate = false;
```

```cpp
while(!terminate){
    cout << "Enter\n1. Create\n2. Display\n3. Insert\n4. Delete\n5. Search\n6. Exit\n";
    int ch;
    cin >> ch;
    int pos, elem;
    switch(ch){
        case 1:
            create(len);
            break;
        case 2:
            display(len);
            cout << endl;
            break;
        case 3:
            cout << "Enter element to insert and position to insert in" << endl;
            cin >> elem >> pos;
            len = insert(elem, pos, len);
            cout << "The new array is : " << endl;
            display(len);
            cout << endl;
            break;
        case 4:
            cout << "Enter the element pos to delete: ";
            cin >> pos;
            if(pos >= len || pos < 0){
                cout << "Operation Not allowed!" << endl;
                break;
            }
            len = delete_elem(pos, len);
            cout << "New array is : ";
            display(len);
            cout << endl;
            break;
        case 5:
          {
            cout << "Enter element to search" << endl;
```

```
                cin >> elem;
                int index = linear_search(len, elem);
                if(index < 0){
                    break;
                }
                cout << "Element found at " << index << endl;
                display(index);
                cout << "[" << A[index] << "]" <<  endl;
                break;
               }
            case 6:
                cout << "Thanks for using" << endl;
                return 0;
        }
    }

    return 0;
}
```

OUTPUT -

```
➜  lab1 git:(master) ✗ ./a.out
Enter
1. Create
2. Display
3. Insert
4. Delete
5. Search
6. Exit
1
Enter elemet at:
0: 1
1: 2
2: 3
3: 4
4: 5
5: 6
```

```
6: 7
7: 8
8:
8
9: 9
Enter
1. Create
2. Display
3. Insert
4. Delete
5. Search
6. Exit
2
1, 2, 3, 4, 5, 6, 7, 8, 8, 9,
Enter
1. Create
2. Display
3. Insert
4. Delete
5. Search
6. Exit
3
Enter element to insert and position to insert in
4
5
The new array is :
1, 2, 3, 4, 5, 4, 6, 7, 8, 8, 9,
Enter
1. Create
2. Display
3. Insert
4. Delete
5. Search
6. Exit
4
Enter the element pos to delete: 2
New array is : 1, 2, 4, 5, 4, 6, 7, 8, 8, 9,
Enter
1. Create
2. Display
3. Insert
4. Delete
```

```
5. Search
6. Exit
5
Enter element to search
2
Element found at 1
1, [2]
Enter
1. Create
2. Display
3. Insert
4. Delete
5. Search
6. Exit
6
Thanks for using
```

Q2

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

int len = 6;
int A[6];

void delete_elem(int pos){
	for(int i = pos; i < len; i++){
		A[i] = A[i+1];
	}
	len--;
}

void check_and_remove_duplicate(int index){
	if(index == len-1){
		return;
	}
	for(int i = 0; i < len; i++){
        if(A[i] == A[index] && i!=index){
            delete_elem(i);
        }
    }
}
void display(){
    for(int i = 0; i < len; i++){
        cout << A[i] << endl;
    }
}
int main(){
    cout << "Enter elements" << endl;
    for(int i = 0; i < len; i++){
        cin >> A[i];
    }
    for(int i = 0; i < len; i++){
        check_and_remove_duplicate(i);
    }
    cout << "Final Array: " << endl;
    display();
}
```

OUTPUT

```
➜  lab1 git:(master) ✗ ./a.out
Enter elements
1
2
3
3
3
4
Final Array:
1
2
```

3
4

Q3.

Ans : 1 (Any garbage value) (Any garbage value) (Any garbage value) (Any garbage value)

Q4

(i)

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

int A[10];
int len = 10;

int main(){
    for(int i = 0; i < len; i++){
        cin >> A[i];
    }
    for(int i = len-1; i >= 0; i--){
        cout << A[i] << ",";
    }
    cout << endl;
}
```

OUTPUT :

```
1
2
3
4
5
6
7
8
9
10
```

10,9,8,7,6,5,4,3,2,1

(ii)

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


int** createMatrix(int rows, int cols){
    int** matrix = new int*[rows];
    for(int i = 0; i < rows; i++){
        matrix[i] = new int[cols];
    }
    return matrix;
}

void feedMatrix(int** matrix, int rows, int cols){
    for(int i = 0; i < rows; i++){
        for(int j = 0; j < cols; j++){
            cout << "[" << i << "-" << j << "] : ";
            cin >> matrix[i][j];
        }
    }
}

void displayMatrix(int** matrix, int rows, int cols){
    for(int i = 0; i < rows; i++){
        cout << "| ";
        for(int j = 0; j < cols; j++){
            cout << matrix[i][j] << " ";
        }
        cout << "|" << endl;
    }
    cout << endl;
```

```
}

int main(){
    int rows1, cols1, rows2, cols2;

    cout << "Enter rows and cols for first matrix" << endl;
    cin >> rows1 >> cols1;

    cout << "Enter rows and cols for second matrix" << endl;
    cin >> rows2 >> cols2;

    if(!rows2 == cols1){
        cout << "Operation not permitted!" << endl;
        return -1;
    }

    int** matrix1 = createMatrix(rows1, cols1);
    int** matrix2 = createMatrix(rows2, cols2);

    cout << "Enter elements for Matrix 1" << endl;
    feedMatrix(matrix1, rows1, cols1);

    cout << "Enter elements for Matrix 2" << endl;
    feedMatrix(matrix2, rows2, cols2);

    int** matrix3 = createMatrix(rows1, cols2);

    for(int i = 0; i < rows1; i++){
        for(int j = 0; j < cols2; j++){
            for(int k = 0; k < cols1; k++){
                matrix3[i][j] += matrix1[i][k] * matrix2[k][j];
                cout << matrix1[i][k] << " " << matrix2[k][j] << " " << matrix3[i][j] << endl;
              }
        }
    }

    displayMatrix(matrix3, rows1, cols2);
    delete[] matrix3;
    delete[] matrix2;
    delete[] matrix1;

    return 0;
}
```

OUTPUT:

```
➜  lab1 git:(master) ✗ ./a.out
Enter rows and cols for first matrix
3
3
Enter rows and cols for second matrix
3
3
Enter elements for Matrix 1
[0-0] : 4
[0-1] : 5
[0-2] : 6
[1-0] : 7
[1-1] : 8
[1-2] : 9
[2-0] : 10
[2-1] : 11
[2-2] : 12
Enter elements for Matrix 2
[0-0] : 13
[0-1] : 14
[0-2] : 15
[1-0] : 16
[1-1] : 17
[1-2] : 18
[2-0] : 19
[2-1] : 20
[2-2] : 21
| 246 261 276 |
| 390 414 438 |
| 534 567 600 |
```

(iii)

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

void display(int** matrix, int rows, int cols){
    for(int i = 0; i < rows; i++){
        cout << "| ";
        for(int j = 0; j < cols; j++){
            cout << matrix[i][j] << " ";
        }
        cout << " |" << endl;
    }
}

void feedMatrix(int** matrix, int rows, int cols){
    for(int i = 0; i < rows; i++){
        for(int j = 0; j < cols; j++){
            cout << "[" << i << "-" << j << "]: ";
            cin >> matrix[i][j];
        }
    }
}

int main(){

    int rows, cols;

    cout << "Enter the number of rows and columns" << endl;
    cin >> rows >> cols;

    //int matrix[rows][cols];

    int** matrix = new int*[rows];
    for(int i = 0; i < rows; i++){
        matrix[i] = new int[cols];
    }

    //feedMatrix(matrix, rows, cols);
    for(int i = 0; i < rows; i++){
        for(int j = 0; j < cols; j++){
            matrix[i][j] = i;
        }
    }

    cout << "======Original Matrix======" << endl;
    display(matrix, rows, cols);

    for(int i = 0; i < rows; i++){
        for(int j = i; j < cols; j++){
            int temp = matrix[i][j];
            matrix[i][j] = matrix[j][i];
            matrix[j][i] = temp;
        }
    }

    cout << "======Transposed Matrix======" << endl;
    display(matrix, cols, rows);

    return 0;
}
```

OUTPUT:

```
➜  lab1 git:(master) ✗ ./a.out
Enter the number of rows and columns
3
3
======Original Matrix======
| 0 0 0  |
| 1 1 1 |
| 2 2 2 |
======Transposed Matrix======
| 0 1 2 |
| 0 1 2 |

| 0 1 2 |
```

Q5.

```
#include <iostream>
#include "utils.h"
using namespace std;


int main(){
    int len;
    cout << "Enter the number of elements" << endl;
    cin >> len;

    int* arr = new int[len];
    feedArray(arr, len);

    int elem;
    cout << "Enter element to search for: ";
    cin >> elem;

    int low = 0;
    int high = len - 1;

    bool flag = true;

    while(low < high){
        int mid = (low+high)/2;
        if(arr[mid] == elem){
            cout << "Foudn at " << mid << endl;
            flag = false;
            break;
        }else if(elem > arr[mid]){
            low = mid+1;
        }else if(elem < arr[mid]){
            high = mid-1;
        }
    }

    if(flag){
            cout << "Element not found" << endl;
    }


}
```

OUTPUT:

```
➜  lab1 git:(master) ✗ ./a.out
Enter the number of rows and columns
3
3
======Original Matrix======
| 0 0 0  |
| 1 1 1  |
| 2 2 2  |
======Transposed Matrix======
| 0 1 2  |
| 0 1 2  |
| 0 1 2  |
```

Q6.

utils.h:

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

void feedArray(int* arr, int len){
	for(int i = 0; i < len; i++){
		cout << i << ": ";
		cin >> arr[i];
	}
}

void randomData(int* arr, int len){
	srand((unsigned)time(0));
	for(int i = 0; i < len; i++){
		arr[i] = 1+(rand() % 100);
	}
}

void display(int* arr, int len){
	for(int i = 0; i < len-1; i++){
		cout << arr[i] << ", ";
	}
	cout << arr[len-1] << endl;
}
```

main.cpp:

```
#include <iostream>
#include "utils.h"
```

```cpp
using namespace std;

int main(){
    int len = 10;
    int arr[len];
    randomData(arr, len);

    cout << "======Original Data======" << endl;
    display(arr, len);

    for(int i = 0; i < len - 1; i++){
        for(int j = 0; j < len - i - 1; j++){
            if(arr[j] > arr[j+1]){
                int temp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = temp;
            }
        }
    }

    cout << "======Sorted Data======" <<  endl;
    display(arr, len);

    return 0;
}
```

OUTPUT:

```
➜  lab1 git:(master) ✗ ./a.out
======Original Data======
52, 32, 46, 12, 94, 23, 80, 64, 13, 79
======Sorted Data======
12, 13, 23, 32, 46, 52, 64, 79, 80, 94
```