ASSIGNMENT - 1(A)

Name: AKSHAT JAIMINI

Roll Number: 102103586

Q1.

(All parts of q1 are divided in their respective header files which are then imported into the final program. Although the individual outputs are given after each part for ease of reading giving a collective output)

FINAL PROGRAM FOR Q1.

```
#include <iostream>
#include "lab_2_1_a.h"
#include "lab_2_1_b.h"
#include "lab_2_1_c.h"
#include "lab_2_1_d.h"
#include "lab_2_1_e.h"
using namespace std;

int main(){
  bool exec = false;
  while(!exec){
    cout << "Enter\n1: Diagonal Matrix\n2: Tridiagonal Matrix\n3: Lower Triangular
Matrix\n4: Upper Triangular\n5: Symmetrical\n";
    int ch = 0;
    cin >> ch;
    switch(ch){
      case 1:
        diagonal();
        break;
      case 2:
        tridiagonal();
        break;
```

```
        case 3:
            lowerTriangle();
            break;
        case 4:
            upperTriangle();
            break;
        case 5:
            symm();
            break;
        default:
            cout << "Wrong choice..Quiting!";
            exec = !exec;
            break;
    }
  }
  return 0;
}
```

(a) *lab_2_1_a.h*

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

void diagonal(){
  cout << "Enter the number of rows and columns: " <<  endl;
  int r = 0,c = 0;
  cin >> r >> c;
  int arr[r];
  // Taking Input
  for(int i = 0; i < r; i++){
    for(int j = 0; j < c; j++){
        cout << "[" << i << ":" << j << "]";
        if(i == j){
            cin >> arr[i];
        }else{
            int dummy;
            cin >> dummy;
```

```
        }
    }
  }
  // Printing
  for(int i = 0; i < r; i++){
    cout << "|" ;
    for(int j = 0; j < c; j++){
      if(i == j){
        cout << arr[i] << " ";
      }else{
        cout << 0 << " ";
      }
    }
    cout << "|" << endl;
  }
}
```

OUTPUT:

```
➜  lab2 git:(master) ✗ ./a.out
Enter
1: Diagonal Matrix
2: Tridiagonal Matrix
3: Lower Triangular Matrix
4: Upper Triangular
5: Symmetrical
1
Enter the number of rows and columns:
3
3
[0:0]1
[0:1]0
[0:2]0
[1:0]0
[1:1]2
[1:2]0
```

[2:0]0
[2:1]0
[2:2]3
|1 0 0 |
|0 2 0 |
|0 0 3 |

(b) ***lab_2_1_b.h***

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

void tridiagonal(){
  int r,c;
  cout << "Enter the number of rows and columns" << endl;
  cin >> r >> c;
  int arr[r+r-1+r-1];
  int counter = 0;
  // Input
  for(int i = 0; i < r; i++){
     for(int j = 0; j < c; j++){
        cout << "[" << i << ":" << j << "] : ";
        if(i == j || i-1==j || i+1==j){
           cin >> arr[counter];
           counter++;
        }else{
           int dummy;
           cin >> dummy;
        }
     }
  }
  counter = 0;
  // Preview
  for(int i = 0; i < r; i++){
     cout << "|" << " ";
     for(int j = 0; j < c; j++){
        if(i == j || i-1 == j || i+1 == j){
           cout << " " << arr[counter] << " ";
           counter++;
        }else{
           cout << " " << 0 << " ";
        }
     }
     cout << "|" << endl;
  }
```

```
}
```

OUTPUT:

```
➜  lab2 git:(master) ✗ ./a.out
Enter
1: Diagonal Matrix
2: Tridiagonal Matrix
3: Lower Triangular Matrix
4: Upper Triangular
5: Symmetrical
2
Enter the number of rows and columns
4
4
[0:0] : 1
[0:1] : 4
[0:2] : 0
[0:3] : 0
[1:0] : 4
[1:1] : 3
[1:2] : 1
[1:3] : 0
[2:0] : 0
[2:1] : 2
[2:2] : 3
[2:3] : 4
[3:0] : 0
[3:1] : 0
[3:2] : 1
[3:3] : 3
|  1  4  0  0 |
|  4  3  1  0 |
|  0  2  3  4 |
|  0  0  1  3 |
```

(c) ***lab_2_1_c.h***

```
// Lower Triangular
#include <iostream>
using namespace std;

void lowerTriangle(){
  int r,c;
  cout << "Enter the number of rows and columns" << endl;
  cin >> r >> c;
  int arr[r+r-1+r-1];
  int counter = 0;
  // Input
  for(int i = 0; i < r; i++){
     for(int j = 0; j < c; j++){
        cout << "[" << i << ":" << j << "] : ";
        if(i >=j ){
           cin >> arr[counter];
           counter++;
        }else{
           int dummy;
           cin >> dummy;
        }
     }
  }
  counter = 0;
  // Preview
  for(int i = 0; i < r; i++){
     cout << "|" << " ";
     for(int j = 0; j < c; j++){
        if(i >= j){
           cout << " " << arr[counter] << " ";
           counter++;
        }else{
           cout << " " << 0 << " ";
        }
     }
     cout << "|" << endl;
```

```
  }
}
```

OUTPUT:

```
➜  lab2 git:(master) ✗ ./a.out
Enter
1: Diagonal Matrix
2: Tridiagonal Matrix
3: Lower Triangular Matrix
4: Upper Triangular
5: Symmetrical
3
Enter the number of rows and columns
3
3
[0:0] : 1
[0:1] : 0
[0:2] : 0
[1:0] : 2
[1:1] : 3
[1:2] : 0
[2:0] : 4
[2:1] : 5
[2:2] : 6
|  1  0  0 |
|  2  3  0 |
|  4  5  6 |
```

(d) *lab_2_1_d.h*

```
// Upper Triangular
#include <iostream>
using namespace std;

void upperTriangle(){
  int r,c;
  cout << "Enter the number of rows and columns" << endl;
  cin >> r >> c;
  int arr[r+r-1+r-1];
  int counter = 0;
  // Input
  for(int i = 0; i < r; i++){
     for(int j = 0; j < c; j++){
        cout << "[" << i << ":" << j << "] : ";
        if(i <=j ){
           cin >> arr[counter];
           counter++;
        }else{
           int dummy;
           cin >> dummy;
        }
     }
  }
  counter = 0;
  // Preview
  for(int i = 0; i < r; i++){
     cout << "|" << " ";
     for(int j = 0; j < c; j++){
        if(i <= j){
           cout << " " << arr[counter] << " ";
           counter++;
        }else{
           cout << " " << 0 << " ";
        }
     }
```

```
    cout << "|" << endl;
  }
}
```

OUTPUT:

```
➜  lab2 git:(master) ✗ ./a.out
Enter
1: Diagonal Matrix
2: Tridiagonal Matrix
3: Lower Triangular Matrix
4: Upper Triangular
5: Symmetrical
4
Enter the number of rows and columns
4
4
[0:0] : 5
[0:1] : 6
[0:2] : 8
[0:3] : 7
[1:0] : 0
[1:1] : 3
[1:2] : 2
[1:3] : 1
[2:0] : 0
[2:1] : 0
[2:2] : 3
[2:3] : 4
[3:0] : 0
[3:1] : 0
[3:2] : 0
[3:3] : 8
|  5  6  8  7 |
|  0  3  2  1 |
|  0  0  3  4 |
```

(e) ***lab_2_1_e.h***

```
// Symmetrical Triangular
#include <iostream>
using namespace std;

void symm(){
  int r,c;
  cout << "Enter the number of rows and columns" << endl;
  cin >> r >> c;
  int arr[r*(c-1)];
  cout << r*(c-1);
  int counter = 0;
  bool x = true;
  // Input
  for(int i = 0; i < r; i++){
     for(int j = 0; j < c; j++){
        cout << "[" << i << ":" << j << "] : ";
        if(i <=j ){
           cin >> arr[counter];
           counter++;
        }else{
           int dummy;
           cin >> dummy;
        }
     }
  }
  cout << "Stored as" << endl;
  for(int i = 0; i < counter; i++){
     cout << arr[i] << " ";
  }
  cout << endl;
}
```

OUTPUT

```
➜  lab2 git:(master) ✗ ./a.out
```

Enter
1: Diagonal Matrix
2: Tridiagonal Matrix
3: Lower Triangular Matrix
4: Upper Triangular
5: Symmetrical
5
Enter the number of rows and columns
4
4
12[0:0] : 2
[0:1] : 4
[0:2] : 6
[0:3] : 0
[1:0] : 4
[1:1] : 1
[1:2] : 9
[1:3] : 5
[2:0] : 6
[2:1] : 9
[2:2] : 4
[2:3] : 7
[3:0] : 0
[3:1] : 5
[3:2] : 7
[3:3] : 0
Stored as
2 4 6 0 1 9 5 4 7 0

Q2.

(All parts of q2 are divided in their respective header files which are then imported into the final program. Although the individual outputs are given after each part for ease of reading giving a collective output)

FINAL PROGRAM FOR Q2.

```
#include <iostream>
#include "lab_2_2_a.h"
#include "lab_2_2_b.h"
#include "lab_2_2_c.h"
using namespace std;

void getArray(int* arr, int r, int c){
  int counter = 0;
  for(int i = 0; i < r; i++){
     for(int j = 0; j < c; j++){
        int x;
        cout << "[" << i << ":" << j << "]" << "->";
        cin >> x;
        if(x!=0){
           arr[counter] = i;
           arr[counter+1] = j;
           arr[counter+2] = x;
           counter += 3;
        }
     }
  }
  // Preview Array
  counter = 0;
  for(int i = 0; i < r; i++){
     cout << "|";
     for(int j = 0; j < c; j++){
        if(arr[counter] == i && arr[counter+1] == j){
           cout << " " << arr[counter+2] << " ";
```

```
            counter += 3;
        }else{
            cout << " " << 0 << " ";
        }
    }
    cout << "|" << endl;
  }
}

int main(){
  bool run = true;
  while(run){
    int ch;
    cout << "Enter:\n1. Transpose\n2. Add\n3. Multiply\n Any Key to quit\n";
    cin >> ch;
    int r,c;
    int arr[300];
    int arr2[300];
    switch(ch){
        case 1:
            cout << "Enter the number of rows and columns" << endl;
            cin >> r >> c;
            getArray(arr,r,c);
            transpose(arr, r,c);
            break;
        case 2:
            cout << "Enter the number of rows and columns" << endl;
            cin >> r >> c;
            getArray(arr,r,c);
            getArray(arr2,r,c);
            add(arr,arr2,r,c);
            break;
        case 3:
            int r,c;
            cout << "Enter the number of rows and columns" << endl;
            cin >> r >> c;
            getArray(arr,r,c);
```

```
            getArray(arr2,r,c);
            multiply(arr,arr2,r,c,r,c);
            break;
        default:
            cout << "Wrong choice.. Exiting!";
            run = false;
            break;
    }
  }
}
```

(a) *lab_2_2_a.h*

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

void transpose(int* arr, int r, int c){
  cout << "Transpose" << endl;
  for(int i = 0; i < c; i++){
    cout << "|";
    for(int j = 0; j < r; j++){
      int counter = 0;
      while(true){
        if(arr[counter] == j && arr[counter+1] == i){
          cout << arr[counter+2] << " ";
          break;
        }
        counter += 3;
      }
    }
    cout << "|" << endl;
  }
}
```

OUTPUT

```
➜  lab2 git:(master) ✗ ./a.out
```

Enter:
1. Transpose
2. Add
3. Multiply
Any Key to quit
1
Enter the number of rows and columns
3
3
[0:0]->1
[0:1]->2
[0:2]->3
[1:0]->4
[1:1]->5
[1:2]->6
[2:0]->7
[2:1]->8
[2:2]->9
| 1 2 3 |
| 4 5 6 |
| 7 8 9 |
Transpose
|1 4 7 |
|2 5 8 |
|3 6 9 |

(b) ***lab_2_2_b.h***

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

void add(int* arr1, int* arr2, int r, int c){
  int counter1 = 0;
  int counter2 = 0;
  int arr[3*r*c];
  int counter3 = 0;
  for(int i = 0; i < r; i++){
    for(int j = 0; j < c; j++){
      int elem1 = 0;
      int elem2 = 0;
      if(arr1[counter1] == i && arr1[counter1+1] == j){
        elem1 = arr1[counter1+2];
        counter1 += 3;
      }
      if(arr2[counter2] == i && arr2[counter2+1] == j){
        elem2 = arr2[counter2+2];
        counter2 += 3;
      }
      cout << ((elem1 + elem2) != 0) << endl;
      if(elem1+elem2 != 0){
        arr[counter3] = i;
        arr[counter3+1] = j;
        arr[counter3+2] = elem1+elem2;
        counter3 +=3;
      }
    }
  }

  counter3 = 0;
  for(int i = 0; i < r; i++){
    cout << "|";
    for(int j = 0; j < c; j ++){
      if(arr[counter3] == i && arr[counter3+1] == j){
```

```
            cout << " " << arr[counter3+2] << " ";
            counter3 += 3;
        }else{
            cout << " " << 0 << " ";
        }
    }
    cout << "|" << endl;
  }
}
```

OUTPUT

```
➜  lab2 git:(master) ✗ ./a.out
Enter:
1. Transpose
2. Add
3. Multiply
Any Key to quit
2
Enter the number of rows and columns
3
3
[0:0]->3
[0:1]->3
[0:2]->4
[1:0]->5
[1:1]->6
[1:2]->7
[2:0]->8
[2:1]->9
[2:2]->10
| 3  3  4 |
| 5  6  7 |
| 8  9  10 |
```

[0:0]->1
[0:1]->2
[0:2]->3
[1:0]->4
[1:1]->5
[1:2]->6
[2:0]->7
[2:1]->8
[2:2]->9
| 1 2 3 |
| 4 5 6 |
| 7 8 9 |
Answer:
| 4 5 7 |
| 9 11 13 |
| 15 17 19 |

(c) *lab_2_2_c.h*

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

void multiply(int* arr1, int* arr2, int r1, int c1, int r2, int c2){
  int arr[r1][c2];
  for(int i = 0; i < r1; i++){
     for(int j = 0; j < c2; j++){
        arr[i][j] = 0;
     }
  }
  int counter1 = 0, counter2 = 0;
  for(int i = 0; i < r1; i++){
     for(int j = 0; j < r2; j++){
        for(int k = 0; k < c2; k++){
           int elem1 = 0;
           int elem2 = 0;
           int counter1 = 0;
           int counter2 = 0;
           while(true){
              if(arr1[counter1] == i && arr1[counter1+1] == k){
                 elem1 = arr1[counter1+2];
                 break;
              }
              counter1 += 3;
           }
           while(true){
              if(arr2[counter2] == k && arr2[counter2+1] == j){
                 elem2 = arr2[counter2+2];
                 break;
              }
              counter2 += 3;
           }
           arr[i][j]+= elem1*elem2;
        }
     }
```

```
  }
  cout << "Answer: " << endl;
  for(int i = 0; i < r1; i++){
     cout << "|";
     for(int j = 0; j < c2; j++){
        cout << " " << arr[i][j] << " ";
     }
     cout << "|" << endl;
  }
}
```

OUTPUT

```
➜  lab2 git:(master) ✗ ./a.out
Enter:
1. Transpose
2. Add
3. Multiply
Any Key to quit
3
Enter the number of rows and columns
3
3
[0:0]->
1
[0:1]->2
[0:2]->3
[1:0]->4
[1:1]->5
[1:2]->6
[2:0]->7
[2:1]->8
[2:2]->9
| 1  2  3 |
| 4  5  6 |
```

| 7 8 9 |
[0:0]->1
[0:1]->2
[0:2]->3
[1:0]->4
[1:1]->5
[1:2]->
6
[2:0]->7
[2:1]->8
[2:2]->9
| 1 2 3 |
| 4 5 6 |
| 7 8 9 |
Answer:
| 30 36 42 |
| 66 81 96 |
| 102 126 150 |

Q3.

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

int rowsum(int i, int c, int** arr){
  int sum = 0;
  for(int j = 0; j < c; j++){
    sum += arr[i][j];
  }
  return sum;
}

int colsum(int j, int r, int** arr){
  int sum = 0;
  for(int i = 0; i < r; i++){
    sum += arr[i][j];
  }
  return sum;
}

void get(int** arr, int r, int c){
  for(int i = 0; i < r; i++){
    cout << "Row " << i << ": " << rowsum(i,r,arr) << endl;
  }
  for(int i = 0; i < c; i++){
    cout << "Col " << i << ": " << colsum(i,r,arr) << endl;
  }
}

int main(){
  int** arr = new int*[4];
  for(int i = 0; i < 4; i++){
    arr[i] = new int[4];
  }
  for(int i = 0; i < 4; i++){
    for(int j = 0; j < 4; j++){
      arr[i][j] = j+i;
```

```
    }
  }
  for(int i = 0; i < 4; i++){
    cout << "|";
    for(int j = 0; j < 4; j++){
      cout << " " << arr[i][j] << " ";
    }
    cout << "|" << endl;
  }
  cout << "OK" << endl;
  get(arr, 4, 4);
  return 0;
}
```

OUTPUT:

```
➜  lab2 git:(master) ✗ ./a.out
| 0  1  2  3 |
| 1  2  3  4 |
| 2  3  4  5 |
| 3  4  5  6 |
OK
Row 0: 6
Row 1: 10
Row 2: 14
Row 3: 18
Col 0: 6
Col 1: 10
Col 2: 14
Col 3: 18
```

Q4.

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

bool checkLowestInRow(int elem, int r, int c, int** arr){
  for(int i = 0; i < c; i++){
     if(elem > arr[r][i]){
        return true;
     }
  }
  return false;
}

bool checkLowestInCol(int elem, int r, int c, int** arr){
  for(int i = 0; i < r; i++){
     if(elem > arr[i][c]){
        return true;
     }
  }
  return false;
}

void feed2D(int** arr, int r, int c){
  srand((unsigned)time(0));
  for(int i = 0; i < r; i++){
     for(int j = 0; j < c; j++){
           arr[i][j] = 1+(rand() % 100);
     }
  }
}

int main(){
  cout << "Enter the number of rows and cols : ";
  int r,c;
  cin >> r >> c;
  int** arr;
```

```
    arr = new int*[r];
    for(int i = 0; i < r; i++){
        arr[i] = new int[c];
    }
    feed2D(arr, r, c);
    cout << "Array is: " << endl;
    for(int i = 0; i < r; i++){
        cout << "|";
        for(int j = 0; j < c; j++){
            cout << " " << arr[i][j] << " ";
        }
        cout << "|" << endl;
    }
    for(int i = 0; i < r; i++){
        for(int j = 0; j < c; j++){
            if(checkLowestInRow(arr[i][j], i, c, arr) && checkLowestInCol(arr[i][j], i, j, arr)){
                cout << "[" << i << ":" << j << "] :" << arr[i][j] << " is a saddle point!" << endl;
            }
        }
    }
    return 0;
}
```

OUTPUT

```
➜  lab2 git:(master) ✗ ./a.out
Enter the number of rows and cols : 4
4
Array is:
| 53  99  88  28 |
| 26  43  66  45 |
| 39  96  94  35 |
| 52  1  13  66 |
[1:3] :45 is a saddle point!
[2:0] :39 is a saddle point!
[2:1] :96 is a saddle point!
[2:2] :94 is a saddle point!
```

[3:0] :52 is a saddle point!

[3:3] :66 is a saddle point!

Q5

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;
}

void feed2D(int** arr, int r, int c){
  srand((unsigned)time(0));
  for(int i = 0; i < r; i++){
	for(int j = 0; j < c; j++){
		arr[i][j] = 1+(rand() % 100);
	}
  }
```

```
}

void display(int** arr, int r, int c){
  for(int i = 0; i < r; i++){
     cout << "|";
     for(int j = 0; j < c; j++){
        cout << arr[i][j] << " ";
     }
     cout << "|" << endl;
  }
}
```

main.cpp

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

int main(){
  int r, c;
  cout << "Enter the number of rows and colums: " << endl;
  cin >> r >> c;
  int** arr = new int*[r];
  for(int i = 0; i < r; i++){
     arr[i] = new int[c];
  }

  feed2D(arr, r, c);
  display(arr, r, c);
  cout << "Spiral traversal " << endl;
  int i, k = 0, l = 0;
  while(k < r && l < c){
     for(i = l; i < c; i++){
        cout << arr[k][i] << endl;
     }
     k++;
     for(i = k; i < r; i++){
```

```
        cout << arr[i][c - 1] << endl;
    }
    c--;
    if(k < r){
        for(i = c - 1; i >= l; i--){
            cout << arr[r-1][i] << endl;
        }
        r--;
    }
    if(l < c){
        for(i = r - 1;i >= k; i--){
            cout << arr[i][l] << endl;
        }
        l++;
    }
  }
}
```

OUTPUT:

➜ lab2 git:(master) ✗ ./a.out

Enter the number of rows and colums:

5

5

|2 86 53 99 53 |

|95 43 55 51 73 |

|20 15 39 51 46 |

|46 70 1 59 29 |

|28 57 82 3 34 |

Spiral traversal

2

86

53

99

53

73

46
29
34
3
82
57
28
46
20
95
43
55
51
51
59
1
70
15
39

Q6

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

int main(){
  int r,c;
  cout << "Enter the number of rows and columns: " << endl;
  cin >> r >> c;
  int arr[r][c];
  int m = r, n = c;
  int k = 0, l = 0 , i = 0;
  while(k < r && l < c){
     for(i = l; i < c; i++){
        cin >> arr[k][i];
     }
     k++;
     for(i = k; i < r; i++){
        cin >> arr[i][c - 1];
     }
     c--;
     if(k < r){
        for(i = c - 1; i >= l; i--){
           cin >> arr[r-1][i];
        }
        r--;
     }
     if(l < c){
        for(i = r - 1;i >= k; i--){
           cin >> arr[i][l];
        }
        l++;
     }
  }
  for(int i = 0; i < m; i++){
     cout << "|";
```

```
    for(int j = 0; j < n; j++){
        cout << " " << arr[i][j] << " ";
    }
    cout << "|" << endl;
  }
}
```

OUTPUT:

```
➜ lab2 git:(master) ✗ ./a.out
Enter the number of rows and columns:
4
4
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
| 1  2  3  4 |
| 12  13  14  5 |
| 11  16  15  6 |
| 10  9  8  7 |
```