1)

Students attendance :

write a program to read the attendance states for 20 students in 15 weeks.

	W0	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13	W14
S0	3	0	0	2	0	0	0	0	2	1	1	0	1	2	2
S1	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0
S2	1	1	1	2	2	2	0	0	0	0	0	0	0	5	4
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
..	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
..	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
S19	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

Then display the following :

- Average absents for each student.
- The student with the maximum absents.
- The student with minimum absents.

```
#include <iostream>
using namespace std;
void main()
{
	int x[20][15];
	cout << "Enter attendance" << endl;
	for (int r = 0; r < 20; r++)
	{
		cout << "Enter for student " << r << endl;
		for (int c = 0; c < 15; c++)
		{
			cin >> x[r][c];
		}
	}

	/*6 days of the week is school*/

	float ct = 6 * 15;

	int max = -99999; int min = 99999; int posx, posn;
	for (int r = 0; r < 20; r++)
	{
		float tot = 0;
		for (int c = 0; c < 15; c++)
		{
			if (x[r][c] != 0)
			{
				tot += 6 - x[r][c];

			}

			if (x[r][c] == 0)
			{
				tot += 6;
			}
		}

		float avg = tot / ct;
```

```
			cout << "Avg absents: " << avg << endl;

				if (tot > max)
				{
					max = tot;
					posx = r;
				}
			if (tot < min)
			{
				min = tot;
				posn = r;
			}
		}
		cout << endl;
		cout << "Student with max absence: student " << posx << endl;
		cout << "Student with min absence: student " << posn << endl;
}
```

6)

- Write a program to read a matrix (100 x 100) from the user.
- Ask the user to select a row.
- Ask the user to select either
  - The right diagonal or
  - The left diagonal.
- Then , swap the selected row with the selected diagonal

e.g.

3	4	5	6
2	4	66	78
90	34	56	79
12	43	67	68

The selected row is : 2
The selected diagonal is: the right

3	4	5	79
2	4	56	78
12	34	66	6
90	43	67	68

```
#include <iostream>
using namespace std;
void main()
{
	int x[100][100], sr,z;
	char diagonal;

	cout << "Enter values (by each row)" << endl;
	for (int r = 0; r < 100; r++)
	{
		for (int c = 0; c < 100; c++)
		{
			cin >> x[r][c];
		}
	}
	cout << endl;
	cout << "Enter row by choice" << endl;
	cin >> sr;
	cout << endl;
	cout << "Enter diagonal by choice (L for left , R for right)" << endl;
	cin >> diagonal;

	if (diagonal == 'L')
	{
		int c = 0;
		for (int r = 0; r < 100; r++)
		{
			z = x[sr][c];
			x[sr][c] = x[r][c];
			x[r][c] = z;
			c++;
		}
	}

	if (diagonal == 'R')
	{
		int c = 99;
		for (int r = 0; r < 100; r++)
```

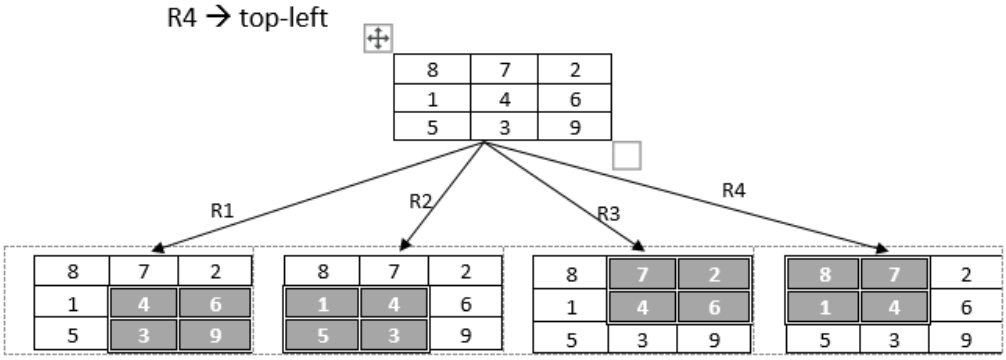
```
                {
                        z = x[sr][c];
                        x[sr][c] = x[r][c];
                        x[r][c] = z;
                        c--;
                }
        }

        for (int r = 0; r < 100; r++)
        {
                for (int c = 0; c < 100; c++)
                {
                        cout << x[r][c] << " ";
                }
                cout << endl;
        }
}
```

3)Rotate Game :

in this game the board (3x3) which started by unsorted numbers (1:9)

the board logically divided into 4 regions

R1 → bottom-right
R2 → bottom-left
R3 → top-right
R4 → top-left



The user will select both : The region & the direction.

e.g.

R1 & Left direction

The user will repeat the steps till the puzzle becomes sorted.

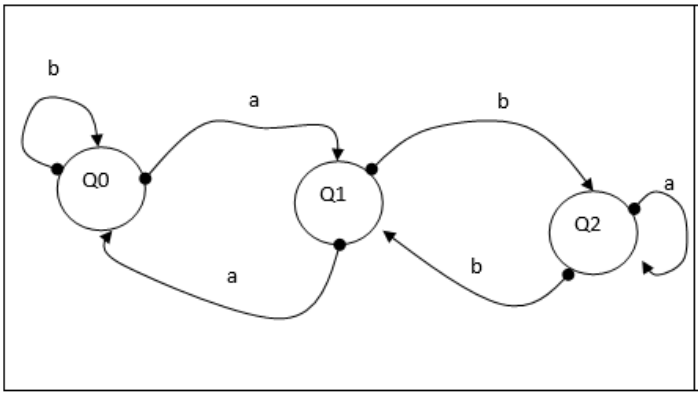
```
#include <iostream>
using namespace std;

void main()
{
	int x[3][3], a, ar, ac, b, br, bc, c, cr, cc, d, dr, dc;
	int region;
	char direction;

	cout << "Enter values (by each row)" << endl;
	for (int r = 0; r < 3; r++)
	{
		for (int c = 0; c < 3; c++)
		{
			cin >> x[r][c];
		}
	}
	cout << endl;


	for (; ;)
	{
		cout << "Enter region (1,2,3,4)" << endl;
	    cin >> region;
```

```
cout << "Enter direction (L,R)" << endl;
cin >> direction;

  if (region == 1)
  {
        a = x[1][1], ar = 1, ac = 1;
        b = x[1][2], br = 1, bc = 2;
        c = x[2][1], cr = 2, cc = 1;
        d = x[2][2], dr = 2, dc = 2;
  }
  if (region == 2)
  {
        a = x[1][0], ar = 1, ac = 0;
        b = x[1][1], br = 1, bc = 1;
        c = x[2][0], cr = 2, cc = 0;
        d = x[2][1], dr = 2, dc = 1;
  }
  if (region == 3)
  {
        a = x[0][1], ar = 0, ac = 1;
        b = x[0][2], br = 0, bc = 2;
        c = x[1][1], cr = 1, cc = 1;
        d = x[1][2], dr = 1, dc = 2;
  }
  if (region == 4)
  {
        a = x[0][0], ar = 0, ac = 0;
        b = x[0][1], br = 0, bc = 1;
        c = x[1][0], cr = 1, cc = 0;
        d = x[1][1], dr = 1, dc = 1;
  }
  if (direction == 'L')
  {
        x[cr][cc] = a;
        x[ar][ac] = b;
        x[br][bc] = d;
        x[dr][dc] = c;

  }
  if (direction == 'R')
  {
        x[br][bc] = a;
        x[ar][ac] = c;
        x[cr][cc] = d;
        x[dr][dc] = b;
  }

  int ct = 0;
  for (int r = 0; r < 3; r++)
  {
        for (int c = 0; c < 2; c++)
        {
              if (x[r][c+1]-x[r][c] ==1 )
              {
                    ct++;
              }
        }
  }
```

```
                if (ct == 6)
                {
                        break;
                }
        }
        for (int r = 0; r < 3; r++)
        {
                for (int c = 0; c < 3; c++)
                {
                        cout << x[r][c] << " ";
                }
                cout << endl;
        }

}
```

4)

Finite State Machine:
Represented either graphically or as matrix



	a	b
	1	0
	0	2
	2	1

Your program will read the matrix that represents a given finite state machine,
Also you have to read a string that contains (a's & b's).
Finally your program will display (accepted in case the machine ended at the final state).

Example:
aabbbab → accepted
aabbbabaab → not accepted

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

void main()
{
	int x[3][2];
	char y[200];

	cout << "Enter values by row" << endl;
	for (int r = 0; r < 3; r++)
	{
		for (int c = 0; c < 2; c++)
		{
			cin >> x[r][c];
		}
	}
	cout << "enter char y" << endl;
	gets_s(y);

	int r = 0;
	int c;
	for (int k = 0; y[k] != '\0'; k++)
	{
		if (y[k] == 'a')
		{
			c = 0;
			r = x[r][c];

		}
```

```
                else
                {
                        c = 1;
                        r = x[r][c];
                }

        }


        if (r == 2)
        {
                cout << "accepted";
        }
        else
        {
                cout << "unaccepted";
        }
}
```

10)

Write a program to read a matrix (20x20) from the user and then do the following:

- Display the triangle with the largest sum.

-1	-2	-3	-1	-2
-3	-6	-2	-5	-4
-5	90	99	-2	-5
-2	99	99	-1	-2
-1	99	99	-1	-2

	0	1	2	3	4
0	-1	**-2**	-3	-1	-2
1	**-3**	**-6**	**-2**	-5	-4
2	-5	90	99	-2	-5
3	-2	99	99	-1	-2
4	-1	99	99	-1	-2

-1	-2	**-3**	-1	-2
-3	**-6**	**-2**	**-5**	-4
-5	90	99	-2	-5
-2	99	99	-1	-2
-1	99	99	-1	-2

-1	-2	-3	**-1**	-2
-3	-6	**-2**	**-5**	**-4**
-5	90	99	-2	-5
-2	99	99	-1	-2
-1	99	99	-1	-2

-1	-2	**-3**	-1	-2
-3	**-6**	**-2**	**-5**	-4
-5	**90**	**99**	**-2**	**-5**
-2	99	99	-1	-2
-1	99	99	-1	-2

-1	-2	-3	-1	-2
-3	**-6**	-2	-5	-4
-5	**90**	**99**	-2	-5
-2	99	99	-1	-2
-1	99	99	-1	-2

-1	-2	-3	-1	-2
-3	-6	**-2**	-5	-4
-5	**90**	**99**	**-2**	-5
-2	99	99	-1	-2
-1	99	99	-1	-2

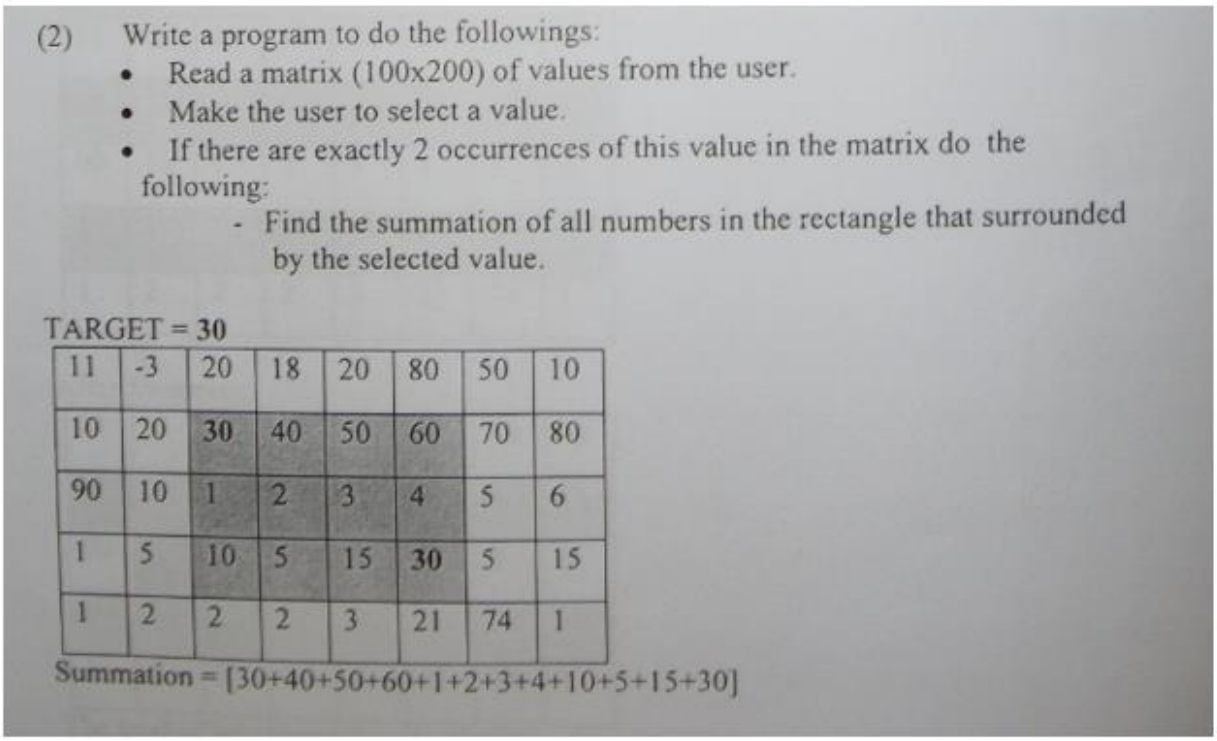
-1	-2	-3	-1	-2
-3	-6	-2	**-5**	-4
-5	90	**99**	**-2**	**-5**
-2	99	99	-1	-2
-1	99	99	-1	-2

-1	-2	-3	-1	-2
-3	-6	**-2**	-5	-4
-5	**90**	**99**	**-2**	-5
-2	**99**	**99**	**-1**	**-2**
-1	99	99	-1	-2

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

void main()
{
	int x[20][20], tot = 0, max = -99999;
	int a, b;
	int posr = 0, posc = 0, limr = 0, limc1 = 0, limc2 = 0;

	cout << "Enter values" << endl;
	for (int r = 0; r < 20; r++)
	{
		for (int c = 0; c < 20; c++)
		{
			cin >> x[r][c];
		}
	}


	for (int r = 0; r < 20; r++)
```

```
	{
		for (int c = 1; c < 19; c++)
		{
			tot = 0;
			tot += x[r][c];
			a = c - 1;
			b = c + 1;
			for (int r1 = r + 1; r1 < 20; r1++)
			{
				if (a - 1 < 0 && r1 + 1 >= 21 || b + 1 > 19 && r1 + 1
>= 21)
				{
					break;
				}
				if (a == 0)
				{
					for (int i = 0; i < 20; i++)
					{
						tot += x[r1][i];
					}
				}
				if (a > 0)
				{
					tot += x[r1][a];
					tot += x[r1][c];
					tot += x[r1][b];
				}
				a--, b++;

				if (tot > max)
				{
					max = tot;
					posr = r;
					posc = c;
				}

			}
		}
	}
	cout << "max:" << max << endl;

	cout << "     " << x[posr][posc] << "     " << endl;
	cout << "   " << x[posr + 1][posc - 1] << " " << x[posr + 1][posc] << " "
<< x[posr + 1][posc + 1] << endl;

	int ct = 1;

	for (int r = posr + 2; r < 20; r++)
	{
		ct++;
		int c;
		for (c = posc - ct; c < posc + ct + 1; c++)
		{
			cout << x[r][c] << " ";
		}
		cout << endl;
		if (c - 1 < 0)
		{
			break;
		}
	}
}
```

11)



(2) Write a program to do the followings:

- Read a matrix (100x200) of values from the user.
- Make the user to select a value.
- If there are exactly 2 occurrences of this value in the matrix do the following:
  - Find the summation of all numbers in the rectangle that surrounded by the selected value.

TARGET = 30

11	-3	20	18	20	80	50	10
10	20	**30**	40	50	60	70	80
90	10	1	2	3	4	5	6
1	5	10	5	15	**30**	5	15
1	2	2	2	3	21	74	1

Summation = [30+40+50+60+1+2+3+4+10+5+15+30]

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

void main()
{
	int x[100][200], ct = 0, ctr = 0, ctc = 0, posr1, posc1, posr2, posc2;

	cout << "Enter values by row" << endl;
	for (int r = 0; r < 100; r++)
	{
		for (int c = 0; c < 200; c++)
		{
			cin >> x[r][c];
		}
	}

	cout << "Enter target:" << endl;
	int target;
	cin >> target;

	for (int r = 0; r < 100; r++)
	{
		for (int c = 0; c < 200; c++)
		{
			if (x[r][c] == target)
			{
				ct++;

				if (ct == 1)
				{
					posr1 = r, posc1 = c;
				}
				if (ct == 2)
```

```
                        {
                                posr2 = r, posc2 = c;
                                break;
                        }
                        if (ct > 2)
                        {
                                ct = 0;
                                posr1 = 0, posr2 = 0, posc1 = 0, posc2 = 0;
                                break;
                        }
                }
        }

    }

    int tot = 0;
    for (int r = posr1; r <=posr2; r++)
    {
        for (int c = posc1; c <= posc2; c++)
        {
            tot += x[r][c];
        }
    }
    cout<<endl;
    cout << "total: " << tot << endl;
}
```