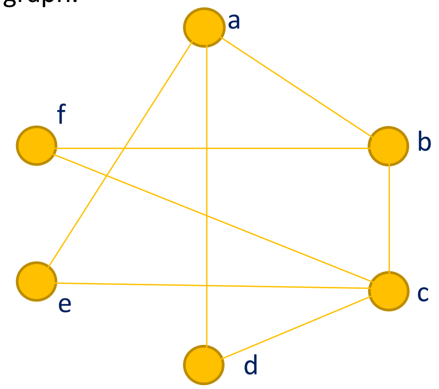
Offline Assignment 1

1) Consider the “adj_list.txt” file. Each line of this file contains two space separated characters (for example: s d) which represents each edge of the following undirected graph.



```
a b
a d
a e
b f
b c
c f
c e
c d
```

Your task is to read both of the characters individually from each line of this file and print the characters in the console using C++ or C. [Use the basic file I/O library] **[4]**

Note: you can’t read the whole line as a string. You must read individual characters separately from each line of the file. To read each character from each line write: yourfilevar >> ch1 >> ch2;

2) In Q(1) you already know how to read each edge information (i.e. each chars) of the graph from the file “adj_list.txt”.

Now build an adjacency list data structure for this graph using STL map of vectors i.e. map<char, vector<char> >

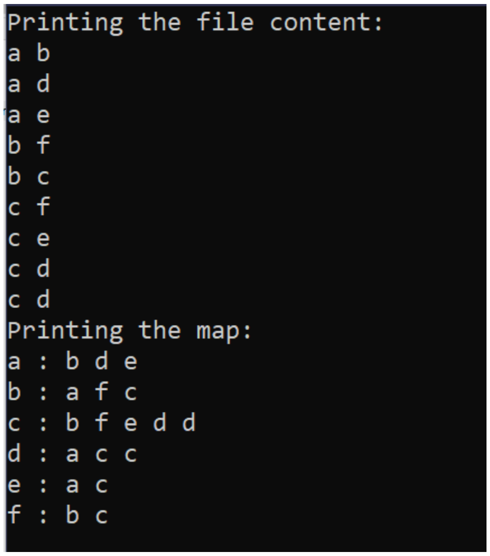
Your map (for example: m) will look like following for the Q(1) graph:

m[a] = [b, d, e]
m[b] = [a, f, c]
m[c] = [f, e, d]
m[d] = [a, c]
m[e] = [a, c]
m[f] = [b, c]

each map cell will contain a vector of characters i.e. vector<char>

Use the solution of Q(1) for reading the file (adj_list.txt) content and then build this map data structure (as shown above) and print the final map in the output console. **[8]**

Q(1) and Q(2) sample output:



```
Printing the file content:
a b
a d
a e
b f
b c
c f
c e
c d
c d
Printing the map:
a : b d e
b : a f c
c : b f e d d
d : a c c
e : a c
f : b c
```

3) We can create a Queue using two stacks (for example, stack s1 and stack s2). **[8]**

Process:

Enqueue operation:

- For queue enqueue/push operation, use the stack (s1) push operation.

Dequeue operation:

- For queue dequeue/pop operation,
    a. First, move all the elements from stack s1 to stack s2. (i.e. pop each element from stack s1 and push that element to s2)
    b. Then print the top element from stack s2. (this top element of s1 is the oldest element)
    c. Then pop the top element from stack s2. (remove this oldest element from stack s2 so our queue dequeue is happening as expected)
    d. Finally, move back all the elements from stack s2 to s1. (i.e. pop each element from stack s2 and push that element to s1)

After implementing this queue push and pop functions using two stacks, push sample integers 10,20,30 to that queue and then pop the front element from this queue and print this popped value.

[Note: you must use 2 stacks for building this queue. For stack you can use STL stack.]

Code Template – use this sample code template. Just read the comments and implement the push and pop functions accordingly. I have already implemented the main() function for you 😉.

```
#include <iostream>
#include <stack>

using namespace std;

stack<int> s1, s2;

///queue push function
void push(int val){
    ///push the value 'val' to stack s1
    /*
        your code
    */
}

///queue pop operation
///this will pop the oldest element from the stack s1 and return the popped item
int pop(){
    ///move all the items from stack s1 to s2
    /*
        your code
    */


    ///store the top element of s2 to a variable
    /*
        your code
    */


    ///pop the oldest item i.e. top item from stack s2
    /*
        your code
    */
    ///move back all the elements from stack s2 to stack s1
    /*
        your code
    */


    ///return the popped value (we have already stored this value in a variable)
    /*
        your code
    */
}

int main()
{
    ///pushing values within the queue
    push(10);
    push(20);
    push(30);

    ///popping the oldest value from the queue and printing it
    cout<< pop() <<endl;

    return 0;
}
```

[Miss at your own risk]

[Any kind of plagiarism is strictly prohibited.]