Sample Assignment 3, Discussed on 2020-09-24, *Not graded*

Question 1 (Storing numberes in a queue) The data structure `std::vector` is very popular; it allows `push_back(...)` method: you can add new elements at the end, but not at the beginning.
Double ended queue (*deque*) is another data structure (as fast and efficient as vector); it allows pushing elements at either end: `push_front(...)` and `push_back(...)`.

File1.cpp

```
1  #include <iostream>
2  #include <string>
3  #include <deque>
4
5  using namespace std;
6  int main() {
7    deque<int> queue1;
8    int n;
9
10   while (cin >> n) {
11     queue1.push_front(n);
12   }
13
14   string sep; // initialized as empty
15   for (int m : queue1) {
16       cout << sep << m; sep = ",";
17   }
18   cout << endl;
19
20   int head = queue1.at(0);
21   queue1.pop_front();
22   cout << head << " -> ";
23   for (int m : queue1) {
24       cout << m << ",";
25   }
26 }
```

Consider the following input on two lines:
input.txt

```
1 3 5
2 4 6
```

What is output from the program to your left on this input?

(A)

```
1,3,5,2,4,6
6 -> 4,2,5,3,1,
```

(B)

```
1,3,5,2,4,6
1 -> 3,5,2,4,6,
```

(C)

```
6,4,2,5,3,1
1 -> 3,5,2,4,6,
```

(D)

```
6,4,2,5,3,1
6 -> 4,2,5,3,1,
```

Question 2 (Reading Line by Line)

Question2.cpp

```
#include <iostream>
#include <sstream>
#include <map>
#include <string>
#include <vector>

using namespace std;
int main() {
  map<int, vector<int>> mymap;

  vector<int> vect;
  string line;
  while (getline(cin, line)) {
    istringstream sstr(line);
    int n;
    while (sstr >> n) {
      vect.push_back(n);
    }

    mymap.insert(make_pair(vect.at(0), vect));
    vect.clear();
  }


  map<int, vector<int>>::iterator it =
      mymap.begin();
  while (it != mymap.end()) {
    cout << it ->first << ": ";
    for (int m : it -> second) {
      cout << m << ",";
    }
    it++; cout << endl;
  }
}
```

Consider the following input on two lines:
input.txt

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

Mark true/false statements about this code:

(A) The output is 3 lines

(B) The output is 1 line

(C) Iterator `it` visits pairs from `mymap` in the same order they were inserted.

(D) Iterator `it` visits pairs from `mymap` in a random order.

(E) Iterator `it` visits pairs from `mymap` in increasing order of the key (`it -> first`).

Question 3: This code inserts some elements in a `std::set`, then tries to find (element `5` is there, but `4` is not). Finally, we iterate over the set in two different ways: as in C++11 (for-loop syntax for an iterator) or an older construct with an explicit iterator.

Question3.cpp

```
1  #include <iostream>
2  #include <set>
3
4  using namespace std;
5  int main() {
6    set<int> myset;
7    myset.insert(11);
8    myset.insert(13);
9    myset.insert(5);
10   myset.insert(7);
11   myset.insert(5);
12   bool b4 = myset.find(4) == myset.end();
13   bool b5 = myset.find(5) == myset.end();
14   cout << "(b4,b5)=(" << b4 << "," <<
15         b5 << ")" << endl;
16   for (int u: myset) {
17       cout << u << "; ";
18   }
19   cout << endl;
20   set<int>::iterator it = myset.begin();
21   while (it != myset.end()) {
22     cout << (*it) << "; ";
23     it++;
24   }
25 }
```

Mark which statements about this code are true/false.

(A) 1st line in output is `(b3,b4) = (0,1)`.

(B) Lines 16–18 and 21–24 iterate over `myset` in the same way.

(C) Iterator on Lines 16–18 visits elements in increasing order.

(D) Iterator on Lines 16–18 visits elements in random order.

Answers

Question 1: D
Question 2: `true`, `false`, `false`, `false`, `true`.
Question 3: `false`, `true`, `true`, `false`.