

Assignment

Lab #3

Course

CIS-7

Section

27168

Due Date

September 12th, 2021

Author

Matthew Mickey

Github repo

https://github.com/PocketPiggy/MVC_Fall_CIS-7/tree/master/Labs/Lab3-Sets

Example #1: myset contains: 5 10 15 20 24 25 26 30 40 50

Problem #1: myset contains: 10 20 24 25 26 30 40 50 60

Observations: 30, 40, 50 is not inserted again.

If keeping 5, 10, 15, the set cardinality would
larger even more so.

```
Alacritty
nvim main.cpp
34     for (it = myset.begin(); it != myset.end(); it++) {
33         cout << " " << *it; // Pointer
32     }
31     cout << endl << endl;
30 }
29
28 void prob2() {
27     set<int> myset;
26     set<int>::iterator it;
25     pair<set<int>::iterator, bool> ret;
24
23     for (int i = 1; i <= 5; i++) {
22         myset.insert(i * 10);
21     }
20
19     ret = myset.insert(20);
18
17     if (!ret.second) {
16         it = ret.first;
15     }
14
13     myset.insert(it, 25);
12     myset.insert(it, 24);
11     myset.insert(it, 26);
10
9     int myints[] = {30, 40, 50, 60}; // Adding needed values
8     myset.insert (myints, myints + 4); // Inserting
7
6     cout << "Problem #1: myset contains:\t";
5     for (it = myset.begin(); it != myset.end(); it++) {
4         cout << " " << *it; // Pointer
3     }
2     cout << endl;
1 }
94
```

Problem #2: theSet contains: 0 2 4 6 8 22 42 68 94
Observations: Elements are inserted in order still, just replacing previous ones.

```
Alacritty
nvim main.cpp
39
38 void prob4() {
37     set<int> theSet;           // Actual set
36     set<int>::iterator itty;  // Iterator
35     pair<set<int>::iterator,bool> ree; // Returns
34
33     for (int i = 0; i <= 8; i += 2) {
32         theSet.insert(i);
31     }
30
29     ree = theSet.insert(42);
28     if (!ree.second) {
27         itty = ree.first;
26     }
25
24     int arrInts[] = {22, 68, 94};
23     theSet.insert(arrInts, arrInts + 3);
22
21     cout << "Problem #2: theSet contains:\t";
20     for (itty = theSet.begin(); itty != theSet.end(); itty++) {
19         cout << " " << *itty;
18     }
17     cout << endl
16     << "Observations: Elements are inserted in order still, just replacing
15     cout << endl << endl;
14 }
13
```

Example #2: myset contains: 13 23 42 65 75

Problem #3 Example #2 CHANGED: myset contains: 13 23 42 55 65

Observation: Changing the element has made the new element sorted/moved into another spot.

```
Alacritty
nvim main.cpp
43
42 void prob5() {
41     int myints[] = {75, 23, 65, 42, 13};
40     set<int> myset(myints, myints + 5);
39
38     cout << "Example #2: myset contains: ";
37     for (set<int>::iterator it = myset.begin(); it !=myset.end(); it++) {
36         cout << " " << *it;
35     }
34     cout << endl;
33 }
32
31 void prob6() {
30     cout << "Problem #3 ";
29     int myints[] = {55, 23, 65, 42, 13};
28     set<int> myset(myints, myints + 5);
27
26     cout << "Example #2 CHANGED: myset contains: ";
25     for (set<int>::iterator it = myset.begin(); it !=myset.end(); it++) {
24         cout << " " << *it;
23     }
22     cout << endl;
21     cout << "Observation: Changing the element has made the new element sorted/
20         << endl << "into another spot.";
19     cout << endl << endl;
18 }
17
```

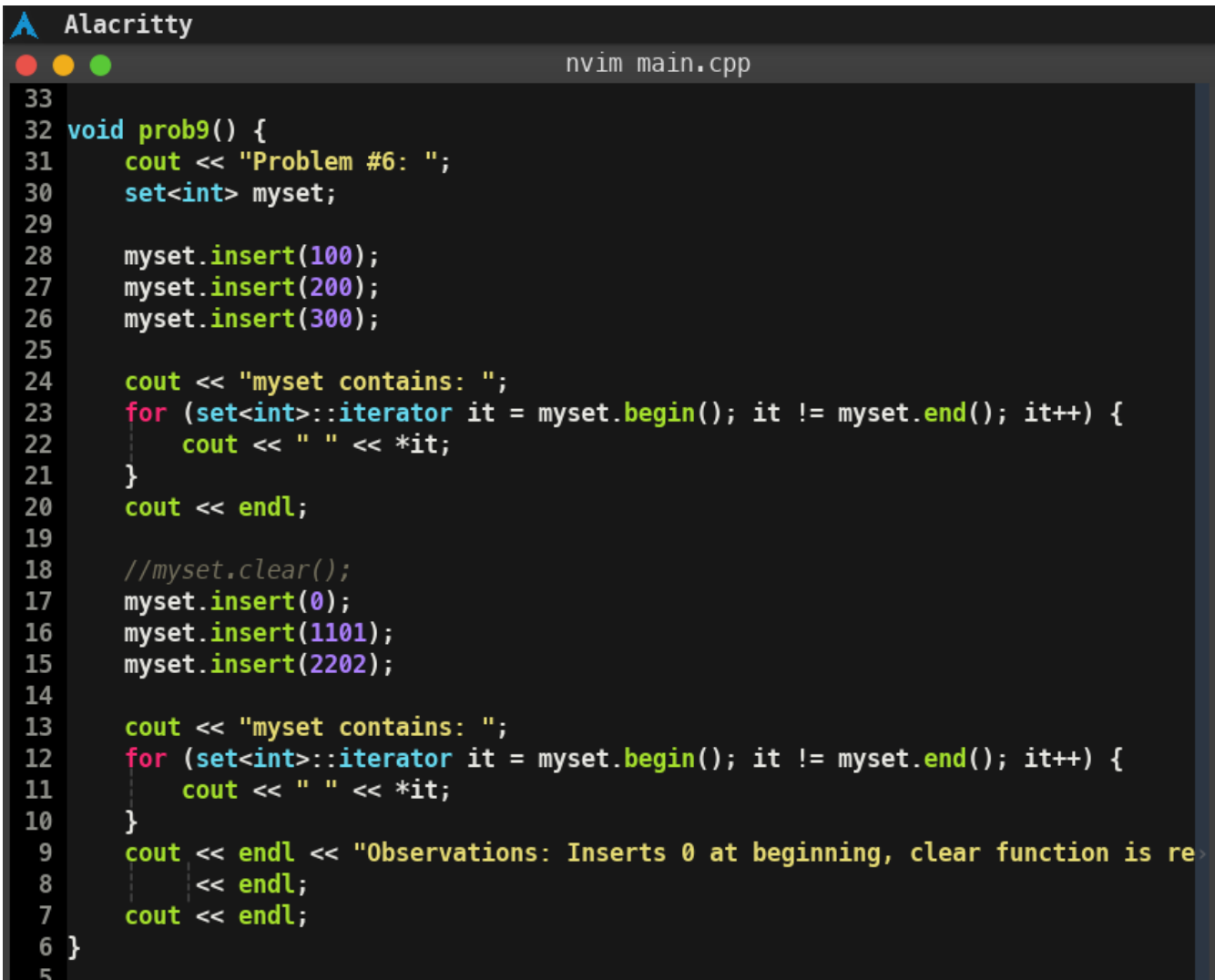
Problem #4: myset contains: -1598737732 94 1575156992
Observation: Addresses are show, even though it's empty set.

```
Alacritty
nvim main.cpp
43 void prob7() {
42     int myints[] = {};
41     set<int> myset(myints, myints + 3);
40
39     cout << "Problem #4: myset contains: ";
38     for (set<int>::iterator it = myset.begin(); it !=myset.end(); it++) {
37         cout << " " << *it;
36     }
35     cout << endl;
34     cout << "Observation: Addresses are show, even though it's empty set.";
33     cout << endl << endl;
32 }
31 }
```

```
Problem #5: myset contains: 100 200 300
myset contains: 1101 2202
Observations: First set of {100, 200, 300} is cleared.
Since myset contains those values first, clearing it would clear those values.
```

```
Alacritty
nvim main.cpp
39 void prob8() {
38     cout << "Problem #5: ";
37     set<int> myset;
36
35     myset.insert(100);
34     myset.insert(200);
33     myset.insert(300);
32
31     cout << "myset contains: ";
30     for (set<int>::iterator it = myset.begin(); it != myset.end(); it++) {
29         cout << " " << *it;
28     }
27     cout << endl;
26
25     myset.clear();
24     myset.insert(1101);
23     myset.insert(2202);
22
21     cout << "myset contains: ";
20     for (set<int>::iterator it = myset.begin(); it != myset.end(); it++) {
19         cout << " " << *it;
18     }
17     cout << endl << "Observations: First set of {100, 200, 300} is cleared." <<
16         << "Since myset contains those values first, clearing it would "
15         << "clear those values.";
14     cout << endl << endl;
13 }
12
```

```
Problem #6: myset contains: 100 200 300
myset contains: 0 100 200 300 1101 2202
Observations: Inserts 0 at beginning, clear function is removed.
```



The image shows a terminal window titled "Alacritty" with a file named "nvim main.cpp" open. The code is a C++ program that demonstrates the use of a set and its operations. It includes a function `prob9()` that initializes a set, inserts elements, prints the set contents, and then inserts additional elements after a commented-out `clear()` call. The program uses `cout` for output and `endl` for new lines. The code is as follows:

```
33
32 void prob9() {
31     cout << "Problem #6: ";
30     set<int> myset;
29
28     myset.insert(100);
27     myset.insert(200);
26     myset.insert(300);
25
24     cout << "myset contains: ";
23     for (set<int>::iterator it = myset.begin(); it != myset.end(); it++) {
22         cout << " " << *it;
21     }
20     cout << endl;
19
18     //myset.clear();
17     myset.insert(0);
16     myset.insert(1101);
15     myset.insert(2202);
14
13     cout << "myset contains: ";
12     for (set<int>::iterator it = myset.begin(); it != myset.end(); it++) {
11         cout << " " << *it;
10     }
9     cout << endl << "Observations: Inserts 0 at beginning, clear function is re
8         << endl;
7     cout << endl;
6 }
5
```

Problem #7 myset contains: 10 20 30

Observation: While condition checks if set is empty or not. If not empty, then print element, then erase it. Which happens to be the first element because the previous one was already erased.

```
19
18 void probl0() {
17     cout << "Problem #7 ";
16     set<int> myset;
15
14     myset.insert(20);
13     myset.insert(30);
12     myset.insert(10);
11
10     cout << "myset contains: ";
9     while (!myset.empty()) {
8         cout << " " << *myset.begin();
7         myset.erase(myset.begin());
6     }
5     cout << endl;
4     cout << "Observation: While condition checks if set is empty or not. If not >
3         << "then print element, then erase it. Which happens to be the first e >
2         << "because the previous one was already erased.";
1     cout << endl;
240
```

NORMAL main.cpp cpp 100% ln:240/240 1 [148]tr...

variable myset