Murdoch UNIVERSITY

Data Structures and Abstractions

Sets & Maps

Lecture 6

Sets

- By definition, Sets are unordered collections of data.
  - A = {2, 1}
  - B = {1, 2, 2, 1, 8/4}
  - C = {x: $x^2 - 3x + 2 = 0$}
  - Note that A = B = C i.e. the sets above are equal to each other [1]
- They are used in maths as well as in many other fields.
- But in the computing domain, there are some variations in the way sets are dealt with.
  - Some sets can contain the actual data values, whilst others only keep a record of the presence or absence of data values. STL Bitsets
  - Elements of a set may not be repeated – a common variation [2]. If repeated elements are needed, then a multiset or bag is used. STL multiset
  - A set is explicitly defined to be unordered, but some implementations require ordering for efficiency reasons [3]. The STL set is an associative container and in STL associative containers are ordered.
  - The last two variations break the Set abstraction but STL designers decided that is fine so sets and multisets are provided. [3]

Sets

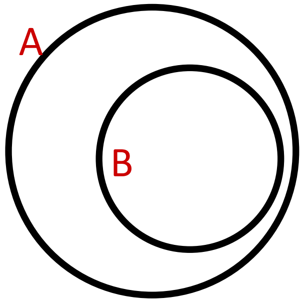
- There are some unique operations for sets:
  - subset
  - union
  - intersection
  - difference
  - element

Subset ⊂

- **Set B is a subset of Set A if all elements in B are also elements of A.**
- **Example 1:**

  if A = {a, b, c, d, g} and B = {c, g}

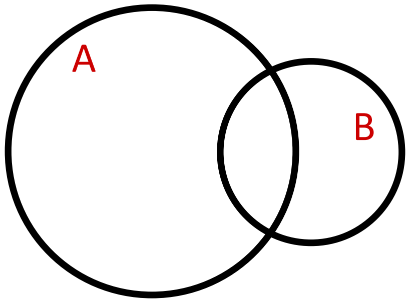
  then B is a subset of A.



- **Example 2:**

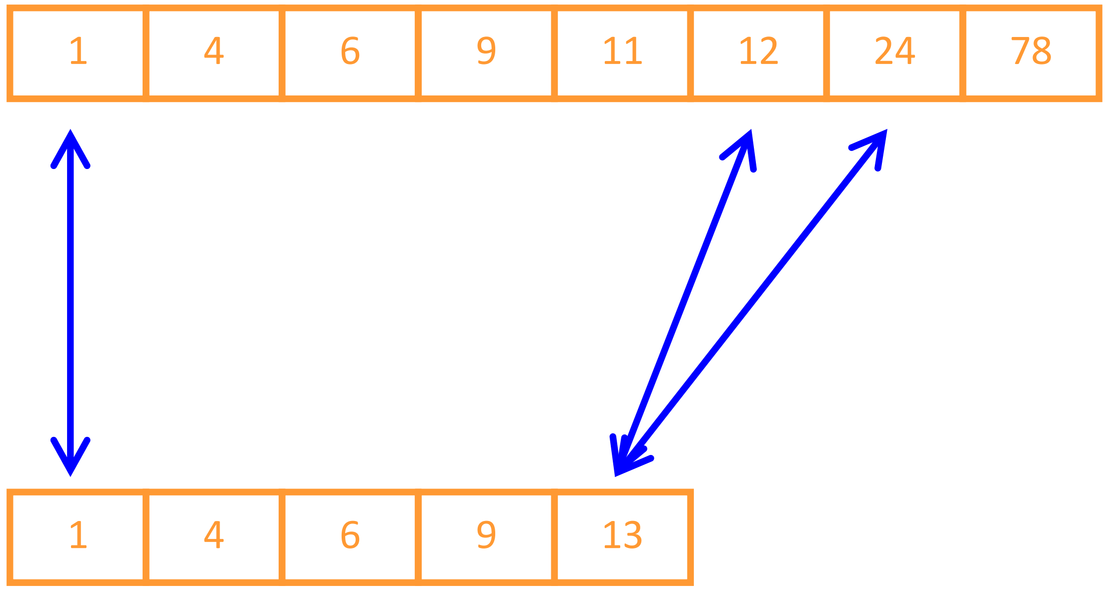
  if A = {a, b, c, d, g} and B = {c, g, u, w}

  then B is not a subset of A.



Subset Animation [1]

subset = true false



END

Subset Pseudo-code

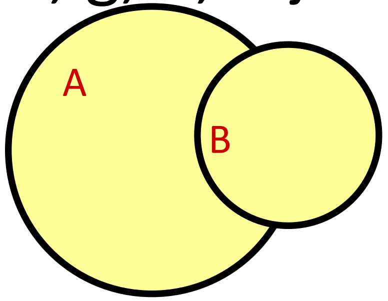
```
IsSubsetOf (other)[1]

    Boolean subset = true
    WHILE more elements in this set AND
             more elements in the other set AND
             subset = true
        IF this element = other element
            Get next element from each set
        ELSE IF this element < other element
            subset = false
        ELSE
            Get next element from other set
        ENDIF
    ENDWHILE
    IF more elements in this set
        subset = false
    ENDIF

END IsSubsetOf
```

Union (or, ||)

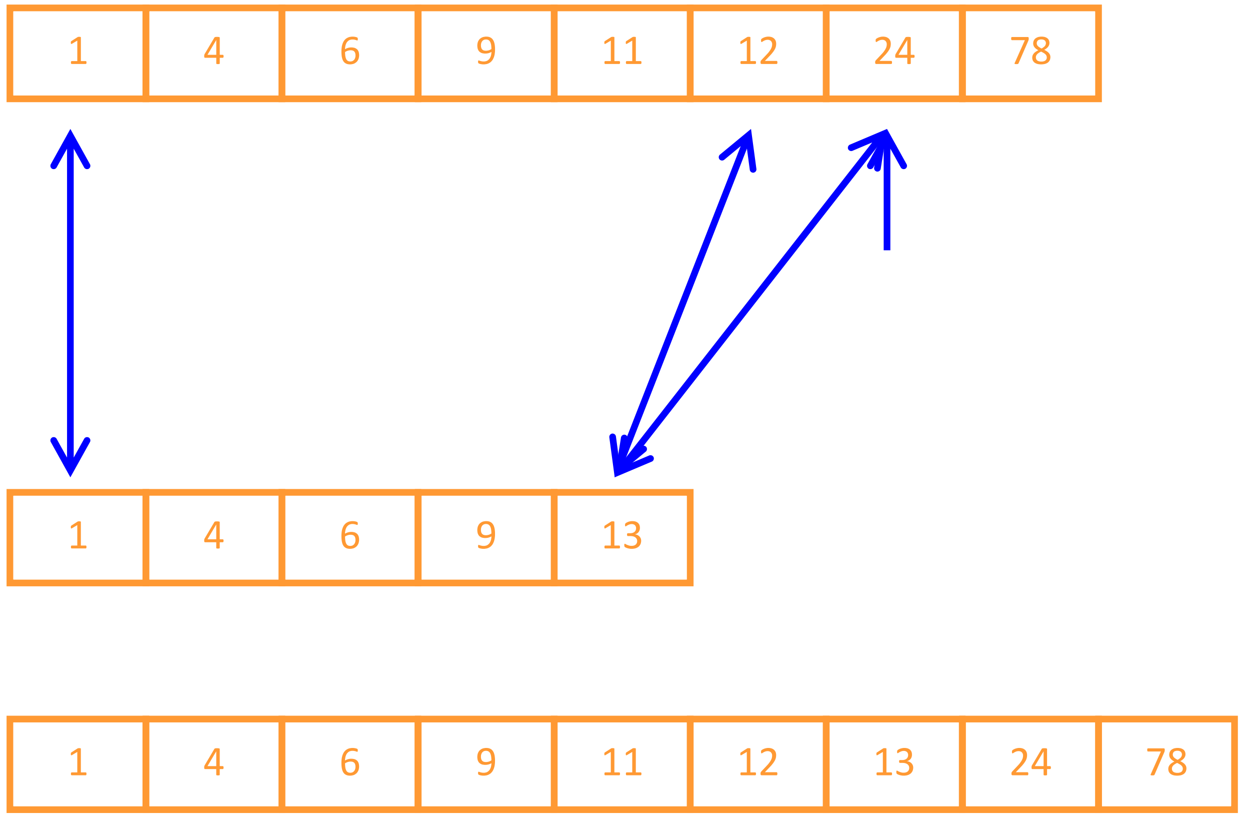
- **The union of Set A and Set B is the collection containing all elements that are in *either* of them, removing double ups. [1]**
- **For example:**

  if A = {a, b, c, d, g} and B = {c, g, u, w}

  then C = A *or* B = {a, b, c, d, g, u, w}
- **C is shown in yellow:**



Union Animation



END

Union Pseudo-code

```
Union (other, newSet) [1]

    WHILE more elements in this set AND
             more elements in the other set
        IF this element = other element
            Add this element into newSet
            Get next element from each set
        ELSE IF this element < other element
            Add this element to newSet
            Get next element from this set
        ELSE
            Add other element to newSet
            Get next element from other set
        ENDIF
    ENDWHILE

    WHILE more elements in this set
        Add this element to newSet
        Get next element from this set
    ENDWHILE

    WHILE more elements in other set
        Add other element to newSet
        Get next element from other set
    ENDWHILE

END Union [2]
```

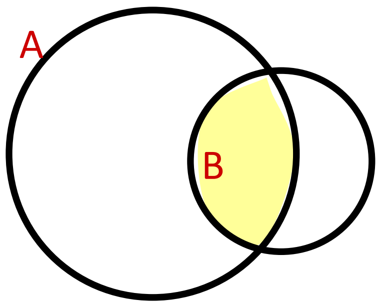
Murdoch UNIVERSITY

Intersection (and, &&)

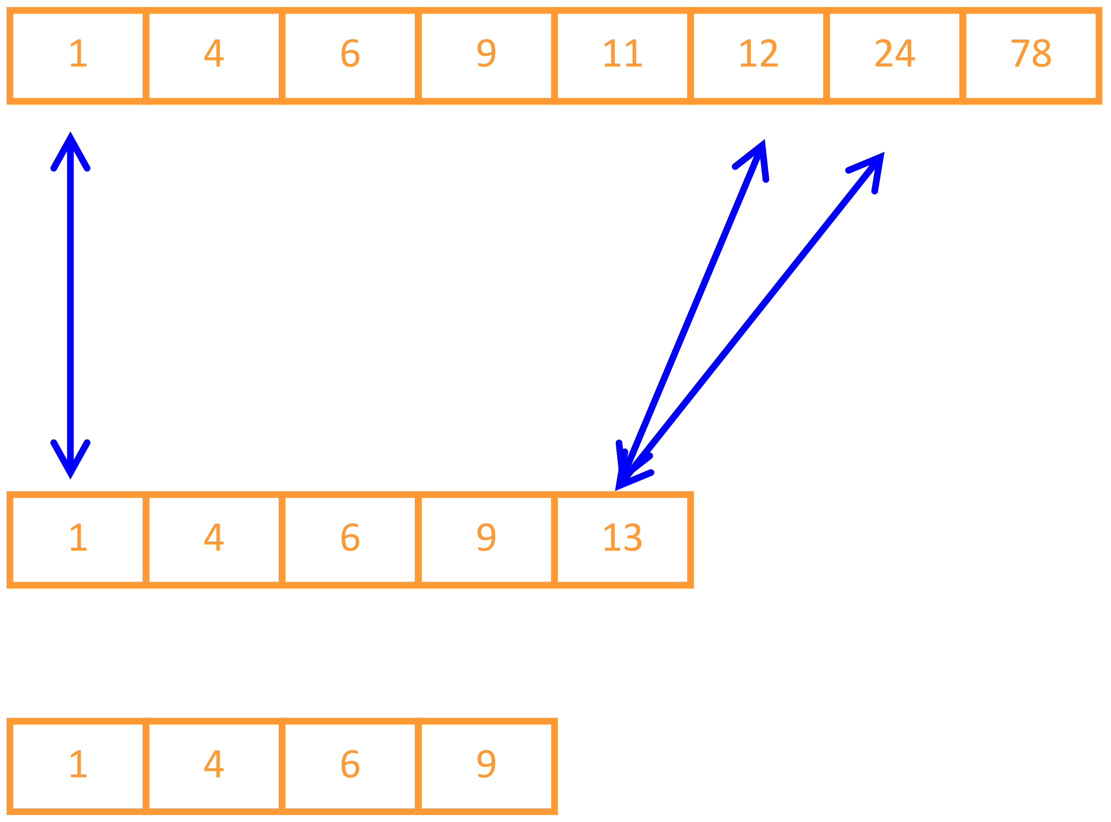
- **The intersection of Set A and Set B is the collection containing all elements that appear in *both* of them.**
- **For example:**

  if A = {a, b, c, d, g} and B = {c, g, u, w}

  then C = A *and* B = {c, g}
- **C is shown in yellow:**



Intersection Animation



END

Intersection Pseudo-code

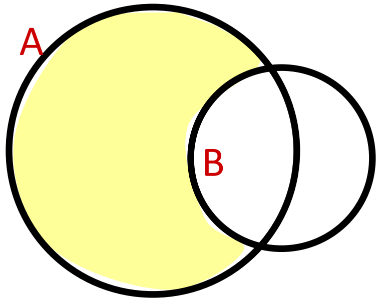
```
Intersection (other, newSet) [1]

     WHILE more elements in this set AND
               more elements in the other set
          IF this element = other element
               Add this element into newSet
               Get next element from each set
          ELSE IF this element < other element
               Get next element from this set
          ELSE
               Get next element from other set
          ENDIF
     ENDWHILE

END Intersecton
```

Difference (-)

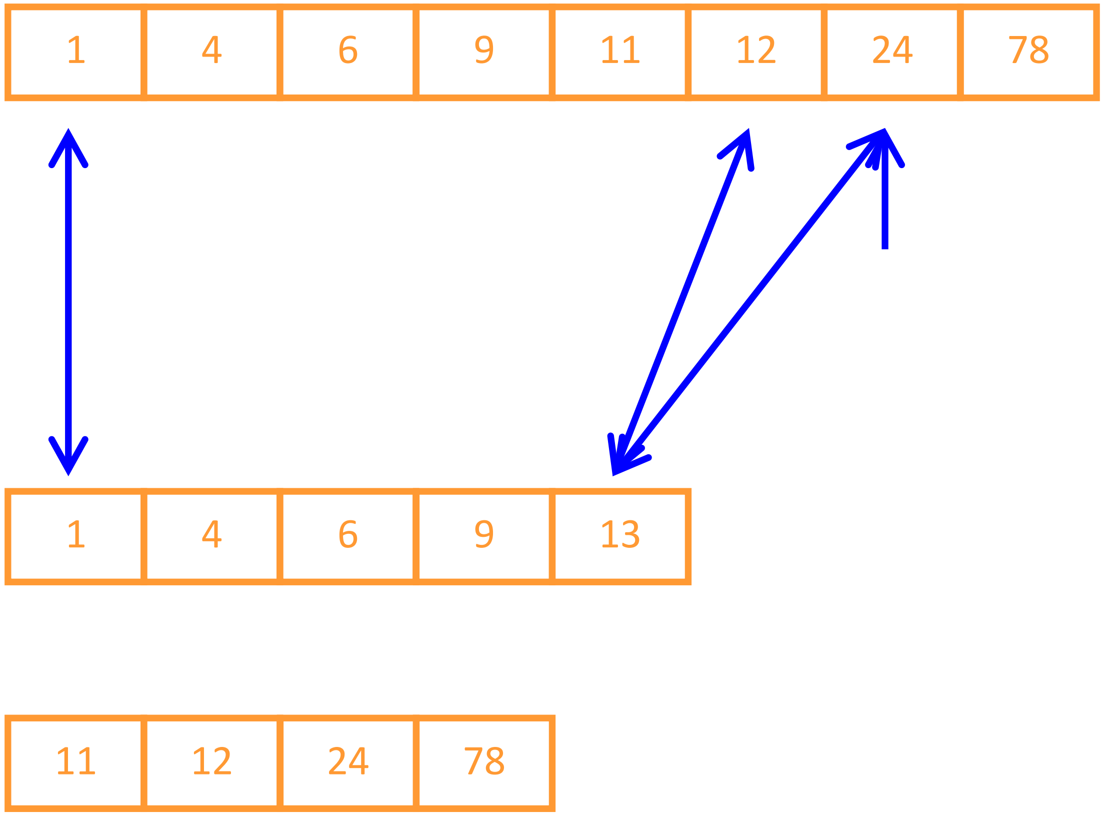
- **The difference of Set B from Set A is the collection containing all elements that are in A but not in B.**
- **For example:**

  if A = {a, b, c, d, g} and B = {c, g, u, w}

  then C = A - B = {a, b, d}
- **C is shown in yellow:**



Difference Animation



END

Difference Pseudo-code

```
Difference(other, newSet) [1]

    WHILE more elements in this set AND
             more elements in the other set
        IF this element = other element
            Get next element from each set
        ELSE IF this element < other element [2]
            Add this element to newSet
            Get next element from this set
        ELSE
            Get next element from other set
        ENDIF
    ENDWHILE
    WHILE more elements in this set
        Add this element to newSet
        Get next element from this container
    ENDWHILE

END Difference
```

The STL Set

- There is an STL set in C++.
- It requires the **`<set>`** header file.
- As for the others it is declared using:

```
typedef set<int> IntSet;
IntSet aset;
```

- The best place to go for information is (again as before):

  http://www.cppreference.com/cppset/index.html

STL Set Methods [1]

`aset.clear()`	Empties the set
`aset.empty()`	Returns true if the set is empty
`aset.begin()`	Returns an iterator to the first element in the set
`aset.end()`	Returns an iterator to past the end of the set
`aset.erase()`	Erase elements from the set
`aset.find()`	Find elements in the set
`aset.insert()`	Insert elements into the set
`aset.size()`	Returns the size of the set
`aset.swap()`	Swaps the contents of two sets

Set Algorithms

- For reasons of general utility, routines that could have been placed in the STL set class were placed in the algorithm class:

```
set_difference(set1.begin(), set1.end(),
               set2.begin(), set2.end());
set_intersection(set1.begin(), set1.end(),
                 set2.begin(), set2.end());
set_union(set1.begin(), set1.end(),
          set2.begin(), set2.end());
```

- These should have been *operations* on sets, because it would have been intuitive.
  - Or (preferred) as helper functions available when #include <set> (What is the Open-closed principle?) [1]
- As these routines have general utility, they can applied to other linear data structures like vectors. This approach can be argued to be good, as re-use of code is happening.
- And there is no subset but a subset set helper function can be written using algorithm's `includes` or `set::find()` function.

```
// Do the set difference using the insert iterator
set_difference(set1.begin(), set1.end(),
               set2.begin(), set2.end(), resultItr);
```

- An abstract representation could have operator-(..) defined, so that:

```
resultSet = set1 - set2
```

- For this reason—unless the task is trivial—the STL set needs to be encapsulated or a helper operator/function is provided.
  - Prefer the helper operator/function as this means the least amount to code for a given functionality.
  - The helper operator or function uses only the set's public interface.

Readings

- Textbook: Standard Template Library, section on Associative containers relating to set and multiset.
- Library EReserve: Preiss, Data structures and algorithms with object-oriented design, Chapter 12
- http://www.cplusplus.com/reference/stl/
- http://en.cppreference.com/w/cpp/container/set
- http://en.cppreference.com/w/Main_Page

MAPS

Note 1 (legacy code only)

- When you compile some STL code in VC++ you might get a warning: [1]

```
ICT283\Code\Sets\SetDifference.cpp(59) : warning C4786:
 'std::pair<std::_Tree<int,int,std::set<int,std::less<int>,std::allocat
 or<int>>::_Kfn,std::less<int>,std::allocator<int>
 >::const_iterator,std::_Tree<int,int,std::set<int,std::less<int>,std::
 allocator<int>>::_Kfn,std::less<int>,std::allocator<int>
 >::const_iterator>' : identifier was truncated to '255' characters in
 the debug information
```

- This is the *only* warning you can ignore completely (a debug identifier)
- If it really annoys you, then add the following code before the includes in the file that is generating the warning:

```
#pragma warning (disable : 4786)
```

- Do *not* disable any other warning!!
  - DO NOT JUST DISABLE THE WARNING IF YOU ARE NOT GETTING THE WARNING.
  - Warning is only on older implementations of Visual C++, so you are not likely to see it now. If you do see it, please let me know. As we
  - **Legacy code and compilers may generate this issue, so just for noting**
  - **You wouldn't see this error in the work you are doing in this unit, but be aware of issues like this with legacy code and older compilers.**

Note 2 (relevant now)

- If you get an error message such as:

```
ICT283\Code\Map\Map.cpp(57) : error C2440: 'initializing' : cannot convert from
'class std::_Tree<class std::basic_string<char,struct
std::char_traits<char>,class std::allocator<char> >,struct std::pair<class
std::basic .....
    No constructor could take the source type, or constructor overload resolution
was ambiguous.
```

- Then it almost always means that you are passing an object as a const reference to a function that uses iterators. Iterators expect references not const references. So, for example, the code below would probably generate this error: [1]

```
void DoSomething (const IntSet &aset) //IntSet is some type with an iterator
                               // typically, this type has had a typedef
                               // typedef set<int> IntSet;
{
      IntSet::iterator itr = aset.begin(); // itr can be used to modify - error
}
```

- To solve it, use a const_iterator, instead of an iterator: [2]

```
void DoSomething (const IntSet &aset)
{


    IntSet::const_iterator itr = aset.begin();
}
```

Maps

- An association (pairing) is a connection between two things, for example the word "sanity" (*key*) is associated with the definition (*value*) "the state of having a normal healthy mind"*
- A dictionary or *map* is then a collection of key-value associations [1].
- The first part of the pair is often called a *key.*
- The data in maps is inserted, deleted and found using the key. So key needs to be unique but value need not be. [2]
- For example, if one had a map that *was* an English dictionary, then we would expect to be able to retrieve the definition of sanity using something like:

```
dictionary.GetDefinition ("sanity");
```

or even

```
dictionary["sanity"];
```

* *Australian Dictionary,* Collins, 2005

The STL Map

- The STL map is a very nice template indeed.
- The declaration requires two data types, the first being the key and the second being the data to be stored in association with the key. [1]
- For example, consider a class taking a vote on who should be the class president. We want to associate names with an integer number of votes:

```
#include <map>
...
map<string, int> Popularity;
...
Popularity pop;
```

A Simple Map Program

```
// Normal comments up here

#include <map>
#include <iostream>
#include <iomanip>
#include <string>

using namespace std; // don't do this – use the approach in the code that is provided separately.

//-----------------------------------------------------

const string END = "end";  // string object

//-----------------------------------------------------

typedef map<string,int> Popularity;
typedef Popularity::iterator PopItr;
typedef Popularity::const_iterator PopCItr; // see textbook chapter on STL
```

It can be really useful to define an iterator for each STL type you use

```
//-------------------------------------------------------

void AddData (Popularity &pop);
void Output  (const Popularity &pop);

//-------------------------------------------------------

int main ()
{
      Popularity pop;

      AddData (pop);
      Output (pop);

      cout << endl;
      return 0;
}

//-------------------------------------------------------
```

```
void AddData (Popularity &pop)
{
        string name;

        // Prime the while loop
        cout << "Enter vote name, or " << END << " to finish: ";
        getline (cin, name);

        while (name != "end") // is this comparison efficient? [1]
        {
                // If they are part of the map already, this adds 1
                //   to their score.  If they are not, it puts them
                //   in the map and gives them a score of 1.
                // see missing code in the notes section [2]

                cout << "Enter vote name, or 'end' to finish: ";
                getline (cin, name);
        }
}
```

Murdoch
UNIVERSITY

```
//----------------------------------------------------

void Output  (const Popularity &pop)
{
          PopCItr winner = pop.begin(); // set a temp winner as the first item

          // For each entry in the map
          for (PopCItr itr = pop.begin(); itr != pop.end(); itr++)
          {
                    // Output the first and second parts of the pair (association)
                    cout << setw(20) << itr->first << " : " << itr->second << endl;

                    // Now check if this person should be the winner
                    if (winner->second < itr->second) // compare the value [1]
                    {
                        winner = itr;
                    }
          }

          // Output the winner
          cout << endl << "The new class president is " << winner->first
                    << " with " << winner->second << " votes" << endl; [2]
}

//----------------------------------------------------
```

Readings

- Textbook: Chapter on Standard Template Library.
- Map: https://en.cppreference.com/w/cpp/container/map
- Multimap: https://en.cppreference.com/w/cpp/container/multimap