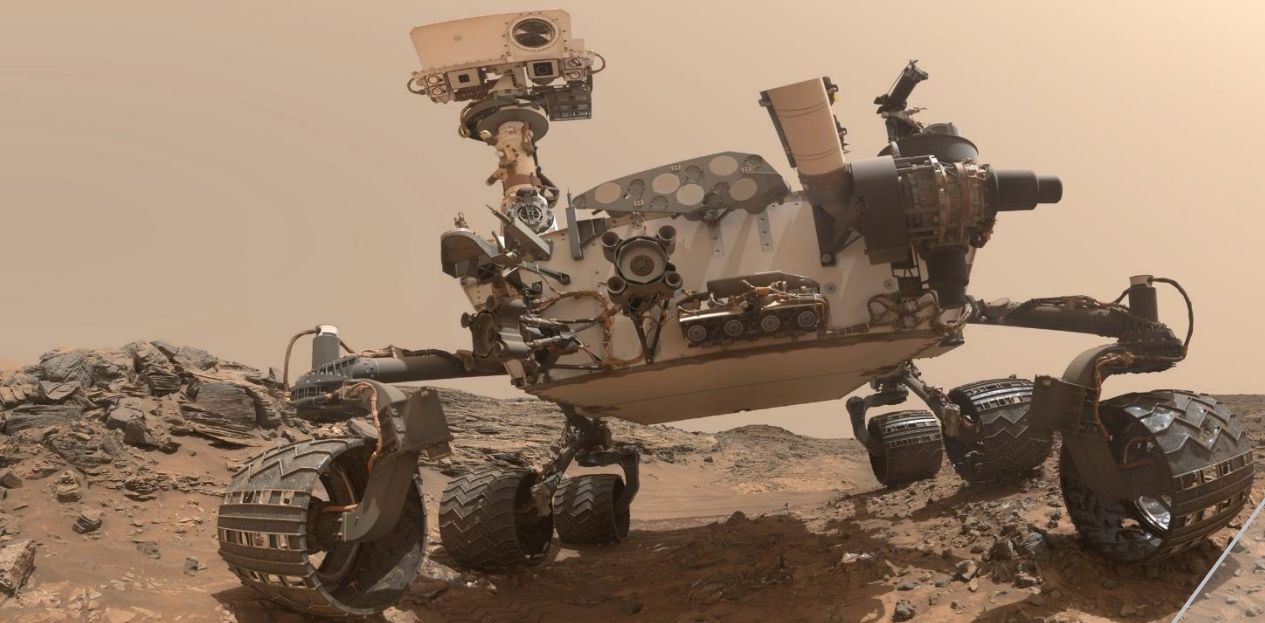
CS101



信奥算法

课件下载地址：
http://pan.baidu.com/s/1o885tz0

作业网站：
http://120.132.18.213:8080/thrall-web/main#home

数据容器

multiset

set

数据容器：multiset和set

multiset 和 set 可较快完成对一组数据的常规操作，包括：

- 插入
- 删除
- 查找
- 计数
- 去重
- 排序
- 找最小/最大

multiset 允许元素重复出现

set 保证元素唯一性，不允许元素重复

set插入和输出

insert()

```
1  #include<iostream>
2  #include<set> //引入set库
3  using namespace std;
4  int main() {
5      set<int> s;  //定义包含整数的set
6      set<int>::iterator it;  //定义迭代器
7      s.insert(8);    //插入元素
8      s.insert(6);
9      s.insert(6);
10     s.insert(6);
11     //循环输出所有元素
12     for(it=s.begin();it!=s.end();it++)
13         cout<<*it<<endl;
14     return 0;
15 }
```

set插入和输出

insert()

```
#include<iostream>
#include<set> //引入set库
using namespace std;
int main() {
    set<int> s;
    set<int>::iter
    s.insert(8);
    s.insert(6);
    s.insert(6);
    s.insert(6);
    //循环输出所有元素
    for(it=s.begin();it!=s.end();it++)
        cout<<*it<<endl;
    return 0;
}
```

自动去重

元素不可以重复出现

自动排序

元素自动从小到大排序

multiset插入和输出 insert()

```
1  #include<iostream>
2  #include<set> //引入set库
3  using namespace std;
4  int main() {
5      multiset<int> s;  //定义包含整数的multiset
6      multiset<int>::iterator it; //定义迭代器
7      s.insert(8);     //插入元素
8      s.insert(6);
9      s.insert(6);
10     s.insert(6);
11     //循环输出set中所有元素
12     for(it=s.begin();it!=s.end();it++)
13         cout<<*it<<endl;
14     return 0;
15 }
```

multiset插入和输出 insert()

```
1  #include<iostream>
2  #include<set> //引入set库
3  using namespace std;
4  int main() {
5      multiset<int> s;
6      multiset<int>::it                t;
7      s.insert(8);      //插入元素
8      s.insert(6);
9      s.insert(6);
10     s.insert(6);
11     //循环输出set中所有元素
12     for(it=s.begin();it!=s.end();it++)
13         cout<<*it<<endl;
14     return 0;
15 }
```

保留重复

元素可以重复出现

自动排序

元素自动从小到大排序

容器，迭代器，从头到尾遍历

```
multiset<int> s;  //定义包含整数的multiset
set<int> s;  //定义包含整数的set

multiset<int>::iterator it; //定义迭代器
set<int>::iterator it;  //定义迭代器
```

迭代器指向元素所在位置，可用于循环遍历

注意符号

set<类型>::iterator

定义，迭代器，从头到尾遍历

```
multiset<int> s;  //定义包含整数的multiset
set<int> s;  //定义包含整数的set

multiset<int>::iterator it; //定义迭代器
set<int>::iterator it;  //定义迭代器
```

```
for(it=s.begin();it!=s.end();it++)
    cout<<*it<<endl;
```

begin()

end()

it是个迭代器指针
*it是it指向的内容

例题：单词排序

输入一行单词序列，相邻单词之间由1个或多个空格间隔，请按照字典序输出这些单词，要求**重复的单词只输出一次**。（区分大小写）数据不含除字母、空格外的其他字符。最多1000个单词

样例输入
She wants to go to Peking University to study Chinese

样例输出
Chinese
Peking
She
University
go
study
to
wants

需要去重

需要排序

例题：单词排序

输入一行单词序列，相邻单词之间由1个或多个空格间隔，请按照字典序输出这些单词，要求**重复的单词只输出一次**。（区分大小写）数据不含除字母、空格外的其他字符。最多1000个单词

样例输入
She wants to go to Peking University to study Chinese

样例输出
Chinese
Peking
She
University
go
study
to
wants

需要去重

自动去重

需要排序

自动排序

例题：单词排序

```
1  #include<iostream>
2  #include<string>
3  #include<set> //引入set库
4  using namespace std;
5  int main() {
6      set<string> s;  //定义包含字符串的set
7      set<string>::iterator it;   //定义迭代器
8      string word;
9      while(cin>>word) s.insert(word);
10     for(it=s.begin();it!=s.end();it++)
11         cout<<*it<<endl;
12     return 0;
13 }
```

set求总量

size()

```
1  #include<iostream>
2  #include<set>
3  using namespace std;
4  int main() {
5      set<int> s;
6      s.insert(1);
7      s.insert(2);
8      s.insert(3);
9      s.insert(3);
10     cout<<s.size(); //求总数
11     return 0;
12 }
```

set删除

erase()

```
1  #include<iostream>
2  #include<set> //引入set库
3  using namespace std;
4  int main() {
5      set<int> s;  //定义包含整数的set
6      set<int>::iterator it;  //定义迭代器
7      s.insert(8);    //插入元素
8      s.insert(6);
9      s.erase(6);     //删除元素
10     s.erase(6);
11     for(it=s.begin();it!=s.end();it++)
12         cout<<*it<<endl;
13     return 0;
14 }
```

multiset删除

erase()

```
#include<iostream>
#include<set> //引入set库
using namespace std;
int main() {
    multiset<int> s;  //定义包含整数的multiset
    multiset<int>::iterator it; //定义迭代器
    s.insert(8);    //插入元素
    s.insert(6); s.insert(6); s.insert(6);
    s.erase(s.find(6));     //删除一个6
    for(it=s.begin();it!=s.end();it++)
        cout<<*it<<' ';
    cout<<endl;
    s.erase(6);     //删除所有6
    for(it=s.begin();it!=s.end();it++)
        cout<<*it<<' ';
    return 0;
}
```

set查找和计数

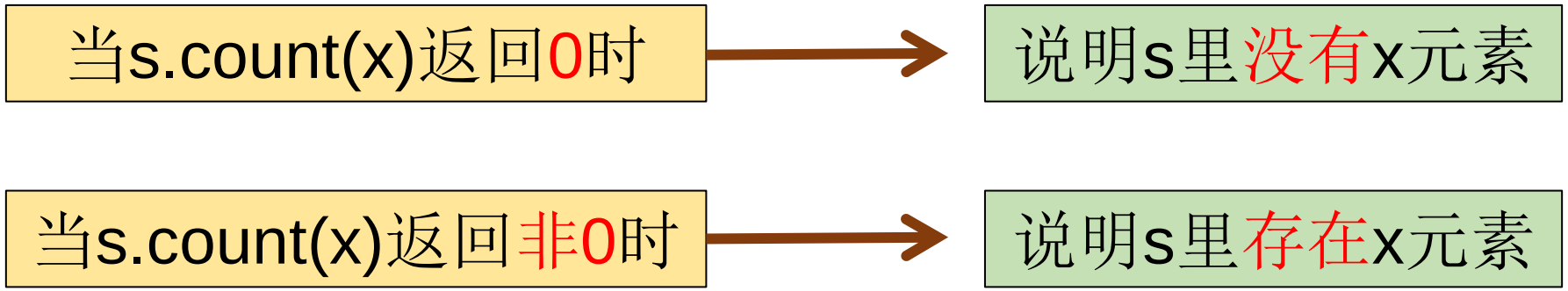
count()

```
1  #include<iostream>
2  #include<set> //引入set库
3  using namespace std;
4  int main() {
5      set<int> s;  //定义包含整数的set
6      set<int>::iterator it;  //定义迭代器
7      s.insert(6);
8      s.insert(6);
9      cout<<s.count(6)<<endl; //返回6的个数
10     cout<<s.count(7)<<endl;
11     return 0;
12 }
```

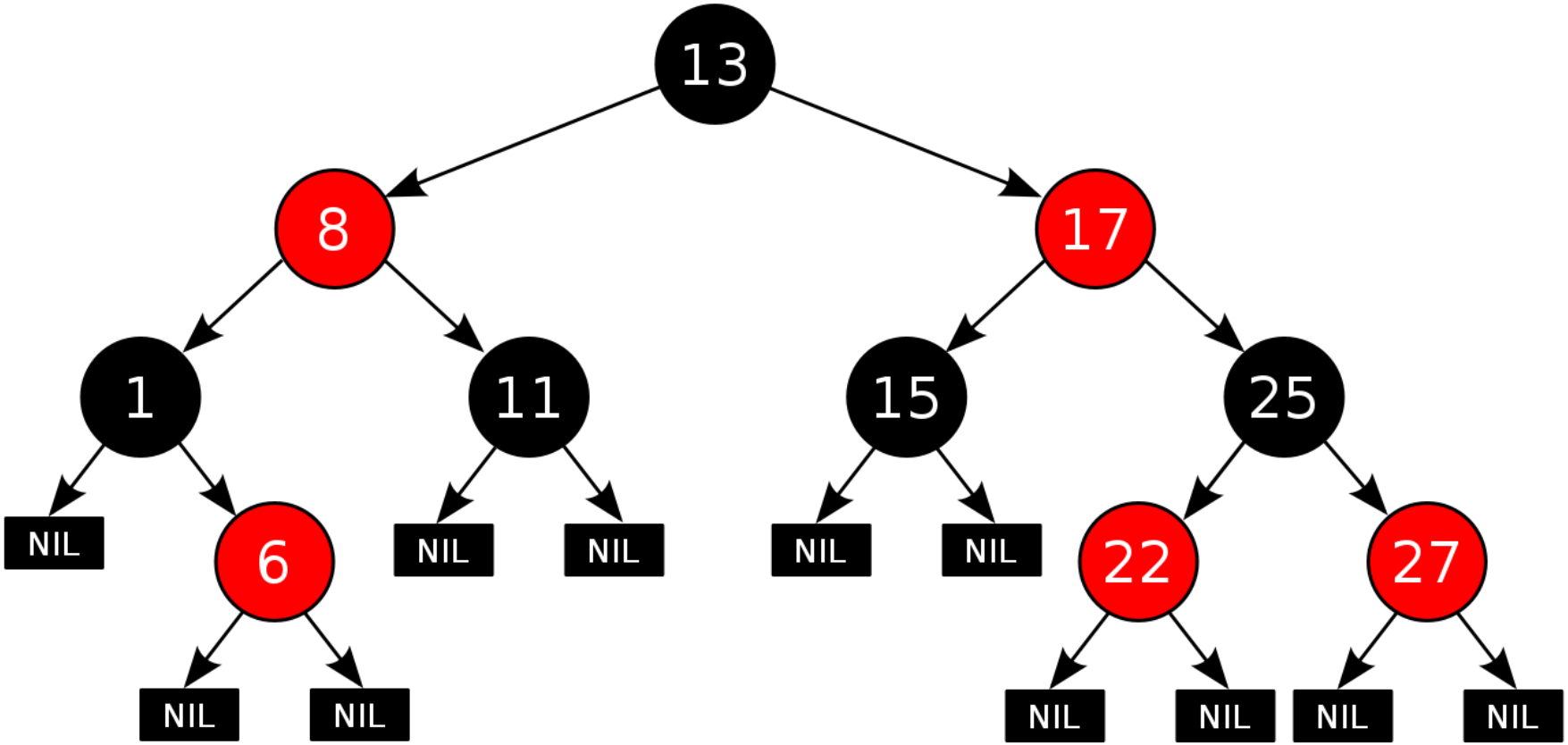
multiset查找和计数 count()

```
1  #include<iostream>
2  #include<set> //引入set库
3  using namespace std;
4  int main() {
5      multiset<int> s;  //定义包含整数的multiset
6      multiset<int>::iterator it; //定义迭代器
7      s.insert(6);
8      cout<<s.count(6)<<endl; //返回6的个数
9      s.insert(6);
10     cout<<s.count(6)<<endl;
11     s.insert(6);
12     cout<<s.count(6)<<endl;
13     cout<<s.count(7)<<endl;
14     return 0;
15 }
```

查找，存在性判断



数据结构：红黑树RB-tree



set和multiset的底层实现都是红黑树

易错点

元素应该**重复**出现时，
不能使用set，
应该使用multiset

大部分情况，
建议使用multiset

multiset 和 set 综合练习

捡袜子

路边捡垃圾的老头正在收集袜子，每只他捡到的袜子都有一种颜色，他希望将相同颜色的袜子配对成为一双袜子。输入第一行为正整数n表示共有几只袜子，第二行为n个小写的英文单词代表每只袜子的颜色，由空格隔开。输出共有几双袜子配对成功。

输入样例
7
red black gold red red black green

输出样例
2

英雄联盟

来自世界各地的英雄人物要组成一个联盟，简称HL。但是HL联盟并不稳定，时常有英雄会因为意见不合离开联盟，也会有新的英雄加入。
输入第一行为正整数m代表有m条关于联盟的信息。以下每一行为加号+或者减号-，以及英雄名字，代表该英雄试图加入或者离开联盟。输出联盟剩余英雄的字典序排列。

输入样例
7
+ironman
+thor
+spiderman
-thor
-thor
-batman
+spiderman

输出样例
ironman
spiderman

僵尸传染

僵尸大战爆发了，最近正常人和僵尸人之间发生了很多互相撕咬的事件。一开始只有一头叫做zombie的僵尸人，僵尸病毒传染性太强，如果正常人被僵尸人咬了一口就会变身成僵尸。输入第一行是m代表有m条互咬信息，以下为m行，按照撕咬先后顺序排列，每行有两个人的小写名字，表示他们两人互相撕咬。输出为最终共有多少个僵尸。

输入样例
5
david mike
mike zombie
zhang mike
wang david
shawn david

输出样例
3

电影推荐

电影爱好者佳佳希望根据她喜欢看的电影，找到另外一些电影，也能符合她口味。为此，她需要知道任意两部电影的相似程度。第一步，她把每部电影加上关键词，第二步，比较两部电影的关键词看看有多少是相同的。

输入有两行代表两部电影，每行冒号前是电影名称，冒号后是一个空格，和一些英文单词，代表这部电影的关键词。输出这两部电影的相似程度，用分数形式表示:a/b，其中整数b表示第二部电影有多少个关键词，整数a表示第二部电影有多少个关键词也在第一部电影里面。

输入样例

World War Z: zombie sci-fi horror infection

I Am Legend: horror sci-fi zombie survivalist infection

输出样例

4/5

说明：第二电影就是I Am Legend共有5个关键词，其中有4个关键词也在第一部电影里面，所以两部电影的相似程度为五分之四。

参考资料

http://www.cplusplus.com/reference/set/set/

http://www.cplusplus.com/reference/set/multiset/