Programming in C/C++

Patrick Ho
peiqistar@gmail.com

(10/4/2014)

STL

- Standard Template Library
- It includes basic algorithms and data structures of computer science
- It includes *container* classes (string, vector, map, queue, stack, set, etc)
- Each class is a C++ template, and can be instantiated to contain any type of object.
- It has common API functions, easy to remember
- It is allowed in USACO and IOI (for general standard version)
- Coding with "using namespace std;", do not need std::

Iterators

- a generalization of pointers, to access data stored in container classes

- To define an iterator variable:
  class_name<template_parameters>::iterator *name*
  e.g. *vector<int>::iterator itr;*

- To get iterator value:
  use class build-in access functions (begin(), end())
  e.g. *itr = myArray.begin();*

- To get class member data, use *
  e.g. *int i2 = (*itr);*

- To get next class member data, use ++
  e.g. *itr++;*

- **There's reverse_interator to be used for backward traverse.**
  e.g. vector<int>::reverse_iterator itr=myArray.rbegin();

Iterators

- **Example: using string**

```
…
string s;
cin >> s;
int sL=s.size();
cout << "size=" << sL<< ‘\n’;
for (int i=0; i<sL; i++)
        cout << s[i]; // use array [ ]
cout << ‘\n’;

// use itr
string::iterator itr = s.begin();
while(itr != s.end()) {              // using while, be careful, itr++
        cout << (*itr);
         itr++;
}
cout << ‘\n’;

// use reverse_iterator
string::reverse_iterator ritr = s.rbegin();
for(; ritr != s.rend(); ritr++)
          cout << (*ritr);
cout << ‘\n’;
```

Vector

- a resizable dynamic array container class

- define a vector variable
  vector<template_parameters> variable_name
  e.g. vector<double> myDArray;

- To add in data member, using push_back()
  e.g. myDArray.push_back(0.5);

- To get the size of array, using size()
  e.g. size_t num=myDArray.size();

- To access data member:
  a. Same as static array using []
  b. Use iterator
  e.g. vector<double>::iterator it = myDArray.begin();

```
for(; it != myDArray.end(); it++)
    double di = (*it);
```

- *To Clean up data, using clear()*

Problem Solving

Decimal to Binary

Write a program to convert positive integer decimal value to a binary value.

Input: data.in file: 1st line is the N number of test values. 2nd to N+1 line has one integer each

Output: Print out the binary value each line

Sample data.in:

5
1
2
5
10
100000

Solution :

Solve using Mod 2 result and save in one vector. Print out in reversed order

Sample output:

1
10
101
1010
11000011010100000

Problem Solving

Solution: (using static array or vector)

```
…
vector<int> vA;
//int iArray[32];
//int iLen;
void convert(int x) {
        vA.clear();
        //Set(iArray, 0);
        //iLen=0;
        while(x>0) {
        vA.push_back(x%2);
        x  = x>>1;
        }
}
void print() {
        vector<int>::reverse_iterator itr=vA.rbegin();
        while(itr!=vA.rend()) {
        printf("%d", (*itr));
        itr++;
        }
        /*
        for(int i=vA.size()-1; i>=0; i--) {
        printf("%d", iArray[i]);
        }
        */
        printf("\n");
}
```

```
int main (int argc, char ** argv)
{
        Rd("data.in");

        int iNum;
        cin >> iNum;
        for(int i=0; i<iNum; i++)
        {
                int iX;
                cin >> iX;
                convert(iX);
                print();
        }

        return 0;
}
```

Map

- map, associates objects of type Key with objects of type Data

- define a map variable
  std::*map*<KEY, *template_parameters*> variable_name
  e.g.
  map<string, int> myIDMap;

- To add in data member, use [ ] and = operator
  e.g.
  myIDMap[“patrick”] = 100;

- To get the size of array, using size()
  e.g.
  size_t num=myIDArray.size();

- Use common STL build-in function (begin(), end(), etc)
- Member has .first (KEY) and .second (DATA)

Map

- To search a Key mamber data, use find()

  e.g.

```
map<string, int>::iterator itr=myIDMap.find(“patrick”);
if (itr == myIDMap.end())
        cout << “ not found” << endl;
else
{
        int myID = (*itr).second;
        cout << myID << endl;
}
```

  * Search speed is a little slow, OK for most small cases

Problem Solving

Search People

Given one person's first name, make a program to find his email address from a yellow book(total people number < 5000).

Input: one file: data.in,

1st line is the total number(N) of yellow book. From 2nd to N+1 line, it has has a person first name string and his email address on each line. The N+2 line is the total number(M) of people to find his email address. From N+3 to N+3+M line, it has a person's first name each line.

Example: data.in

4
Patrick peiqistar@gmail.com
Tim T@yahoo.com
Tom Tom@gmail.com
Jim J@gmail.com
3
Tim
Pak
Tom

Output:

Print out found email address. If not found, print message, "NOT FOUND", each line.

T@yahoo.com
NOT FOUND
Tom@gmail.com

Solution A:

Solve in class together