Final Project Presentation

Bees - C++ Implementation of the Java STL

劉爽

上海交通大學致遠學院 2011 級

2012.5.7

Project Summary

- 利用 C++ 的 template 功能實現部分 JAVA 的 STL 功能
- 在給定一個 Utility.h 頭文件以及其餘六個頭文件的函數定義框架的情況下補全每個函數體內部的內容
- 最終需要實現線性查找表 ArrayList 與 linkedList, 哈希查找表 HashSet 與 HashMap, 樹形查找表 TreeSet 與 TreeMap 六個文件

Project Summary

- 利用 C++ 的 template 功能實現部分 JAVA 的 STL 功能
- 在給定一個 Utility.h 頭文件以及其餘六個頭文件的函數定義框架的情況下補全每個函數體內部的內容
- 最終需要實現線性查找表 ArrayList 與 linkedList, 哈希查找表 HashSet 與 HashMap, 樹形查找表 TreeSet 與 TreeMap 六個文件

Project Summary

- 利用 C++ 的 template 功能實現部分 JAVA 的 STL 功能
- 在給定一個 Utility.h 頭文件以及其餘六個頭文件的函數定義框架的情況下補全每個函數體內部的內容
- 最終需要實現線性查找表 ArrayList 與 linkedList, 哈希查找表 HashSet 與 HashMap, 樹形查找表 TreeSet 與 TreeMap 六個文件

Project Summary

- 利用 C++ 的 template 功能實現部分 JAVA 的 STL 功能
- 在給定一個 Utility.h 頭文件以及其餘六個頭文件的函數定義框架的情況下補全每個函數體內部的內容
- 最終需要實現線性查找表 ArrayList 與 linkedList, 哈希查找表 HashSet 與 HashMap, 樹形查找表 TreeSet 與 TreeMap 六個文件

Work Description - Preparing

- 閱讀任務文檔，熟悉任務要求
- 閱讀 JAVA 文檔及源碼，選擇了 GNU CLASSPATH
- 瞭解 JAVA 實現方法，并研究如果運用在 C++ 中會有什麼困難

Work Description - Preparing

- 閱讀任務文檔，熟悉任務要求
- 閱讀 JAVA 文檔及源碼，選擇了 GNU CLASSPATH
- 瞭解 JAVA 實現方法，幷研究如果運用在 C++ 中會有什麼困難

Work Description - Preparing

- 閱讀任務文檔，熟悉任務要求
- 閱讀 JAVA 文檔及源碼，選擇了 GNU CLASSPATH
- 瞭解 JAVA 實現方法，幷研究如果運用在 C++ 中會有什麼困難

Work Description - Preparing

- 閱讀任務文檔，熟悉任務要求
- 閱讀 JAVA 文檔及源碼，選擇了 GNU CLASSPATH
- 瞭解 JAVA 實現方法，幷研究如果運用在 C++ 中會有什麼困難

Work Description - Implementation - ArrayList

```
class ArrayList {
    private:
    static const int DEFAULT_CAPACITY = 10;
    int sz, cap;
    E* data;
```

```
class Iterator {
    private:
    int pos, size, last;
    ArrayList* arr;
```

```
    void ensureCapacity(int minCapacity) {
        if (minCapacity > cap) {
            E* newData = new E[cap = getMax(cap * 2, minCapacity)];
            memmove(newData, data, sz * sizeof(E));
            delete [] data;
            data = newData;
        }
    }
```

Work Description - Implementation - ArrayList

```
class ArrayList {
    private:
    static const int DEFAULT_CAPACITY = 10;
    int sz, cap;
    E* data;
```

```
class Iterator {
    private:
    int pos, size, last;
    ArrayList* arr;
```

```
    void ensureCapacity(int minCapacity) {
        if (minCapacity > cap) {
            E* newData = new E[cap = getMax(cap * 2, minCapacity)];
            memmove(newData, data, sz * sizeof(E));
            delete [] data;
            data = newData;
        }
    }
```

Work Description - Implementation - ArrayList

```
class ArrayList {
    private:
    static const int DEFAULT_CAPACITY = 10;
    int sz, cap;
    E* data;
```

```
class Iterator {
    private:
    int pos, size, last;
    ArrayList* arr;
```

```
    void ensureCapacity(int minCapacity) {
        if (minCapacity > cap) {
            E* newData = new E[cap = getMax(cap * 2, minCapacity)];
            memmove(newData, data, sz * sizeof(E));
            delete [] data;
            data = newData;
        }
    }
```

Work Description - Implementation - ArrayList

```
class ArrayList {
    private:
    static const int DEFAULT_CAPACITY = 10;
    int sz, cap;
    E* data;
```

```
class Iterator {
    private:
    int pos, size, last;
    ArrayList* arr;
```

```
    void ensureCapacity(int minCapacity) {
        if (minCapacity > cap) {
            E* newData = new E[cap = getMax(cap * 2, minCapacity)];
            memmove(newData, data, sz * sizeof(E));
            delete [] data;
            data = newData;
        }
    }
```

Work Description - Implementation - LinkedList

```
class LinkedList {
    private:
    class Entry {
        public:
        T data;
        Entry *next, *previous;
        Entry() {}
        Entry(T _data) {
            data = _data;
            next = previous = NULL;
        }
    };
    Entry *first, *last;
    int sz;
```

Work Description - Implementation - LinkedList

```
class LinkedList {
    private:
    class Entry {
        public:
        T data;
        Entry *next, *previous;
        Entry() {}
        Entry(T _data) {
            data = _data;
            next = previous = NULL;
        }
    };
    Entry *first, *last;
    int sz;
```

Work Description - Implementation - HashMap

```
class HashMap {
    public:
    static const int DEFAULT_CAPCITY = 11;
    static const double DEFAULT_LOAD_FACTOR = 0.75;
    private:
    template <class K2, class V2>
    class HashEntry: public Entry<K2, V2> {
        public:
        HashEntry<K2, V2>* next;
        HashEntry() {}
        HashEntry(K2 _key, V2 _value): Entry<K2, V2>(_key, _value)
{}
    };
    int threshold, cap;
    double loadFactor;
    HashEntry<K, V>** buckets;
    int sz;
```

Work Description - Implementation - HashMap

```
class HashMap {
    public:
    static const int DEFAULT_CAPCITY = 11;
    static const double DEFAULT_LOAD_FACTOR = 0.75;
    private:
    template <class K2, class V2>
    class HashEntry: public Entry<K2, V2> {
        public:
        HashEntry<K2, V2>* next;
        HashEntry() {}
        HashEntry(K2 _key, V2 _value): Entry<K2, V2>(_key, _value)
{}
    };
    int threshold, cap;
    double loadFactor;
    HashEntry<K, V>** buckets;
    int sz;
```

Work Description - Implementation - HashMap & HashSet

```
V put(const K& key, const V& value) {
    if (++sz > threshold) {
        rehash();
        idx = hash(key);
    }
```

```
void rehash() {
    cap = cap * 2 + 1;
    threshold = (int)(cap * loadFactor);
```

```
class HashSet {
    private:
    HashMap<T, bool, H>* map;

class Iterator {
    public:
    typename HashMap<T, bool, H>::Iterator mItr;
```

Work Description - Implementation - HashMap & HashSet

```
V put(const K& key, const V& value) {
    if (++sz > threshold) {
        rehash();
        idx = hash(key);
    }
```

```
void rehash() {
    cap = cap * 2 + 1;
    threshold = (int)(cap * loadFactor);
```

```
class HashSet {
    private:
    HashMap<T, bool, H>* map;

class Iterator {
    public:
    typename HashMap<T, bool, H>::Iterator mItr;
```

Work Description - Implementation - HashMap & HashSet

```
V put(const K& key, const V& value) {
    if (++sz > threshold) {
        rehash();
        idx = hash(key);
    }
```

```
void rehash() {
    cap = cap * 2 + 1;
    threshold = (int)(cap * loadFactor);
```

```
class HashSet {
    private:
    HashMap<T, bool, H>* map;

class Iterator {
    public:
    typename HashMap<T, bool, H>::Iterator mItr;
```

Work Description - Implementation - HashMap & HashSet

```
V put(const K& key, const V& value) {
    if (++sz > threshold) {
        rehash();
        idx = hash(key);
    }
```

```
void rehash() {
    cap = cap * 2 + 1;
    threshold = (int)(cap * loadFactor);
```

```
class HashSet {
    private:
    HashMap<T, bool, H>* map;

class Iterator {
    public:
    typename HashMap<T, bool, H>::Iterator mItr;
```

Work Description - Implementation - TreeMap

```
class TreeMap {
    private:
    static const int RED = -1, BLACK = 1;
    template <class K2, class V2>
    class Node: public Entry<K2, V2> {
        public:
        int color;
        Node<K2, V2> *left, *right, *parent;
        Node(): Entry<K2, V2>(K(), V()) {
            color = BLACK;
            left = right = parent = this;
        }
        Node(K2 _key, V2 _value, int _color, Node<K2, V2>* _left,
                Node<K2, V2>* _right, Node<K2, V2>* _parent):
                    Entry<K2, V2>(_key, _value)
        {
            color = _color;
            left = _left; right = _right; parent = _parent;
        }
    };
    Node<K, V> *nil, *root; int sz;
```

Work Description - Implementation - TreeMap

```
class TreeMap {
    private:
    static const int RED = -1, BLACK = 1;
    template <class K2, class V2>
    class Node: public Entry<K2, V2> {
        public:
        int color;
        Node<K2, V2> *left, *right, *parent;
        Node(): Entry<K2, V2>(K(), V()) {
            color = BLACK;
            left = right = parent = this;
        }
        Node(K2 _key, V2 _value, int _color, Node<K2, V2>* _left,
                Node<K2, V2>* _right, Node<K2, V2>* _parent):
                        Entry<K2, V2>(_key, _value)
        {
            color = _color;
            left = _left; right = _right; parent = _parent;
        }
    };
    Node<K, V> *nil, *root; int sz;
```

Work Description - Implementation - TreeMap & TreeSet

```
class TreeMap {
    private:
        void rotateLeft(Node<K, V>* node);
        void rotateRight(Node<K, V>* node);
        void insertFixup(Node<K, V>* node);
        void deleteFixup(Node<K, V>* node, Node<K, V> *parent);
        void removeNode(Node<K, V>* node);
        Node<K, V>* getNode(K key);
        Node<K, V>* successor(Node<K, V>* node);
```

```
class TreeSet {
    private:
        TreeMap<E, bool>* map;

class Iterator {
    public:
    typename TreeMap<E, bool>::Iterator mItr;
```

Work Description - Implementation - TreeMap & TreeSet

```
class TreeMap {
    private:
        void rotateLeft(Node<K, V>* node);
        void rotateRight(Node<K, V>* node);
        void insertFixup(Node<K, V>* node);
        void deleteFixup(Node<K, V>* node, Node<K, V> *parent);
        void removeNode(Node<K, V>* node);
        Node<K, V>* getNode(K key);
        Node<K, V>* successor(Node<K, V>* node);
```

```
class TreeSet {
    private:
        TreeMap<E, bool>* map;

class Iterator {
    public:
    typename TreeMap<E, bool>::Iterator mItr;
```

Work Description - Implementation - TreeMap & TreeSet

```
class TreeMap {
    private:
        void rotateLeft(Node<K, V>* node);
        void rotateRight(Node<K, V>* node);
        void insertFixup(Node<K, V>* node);
        void deleteFixup(Node<K, V>* node, Node<K, V> *parent);
        void removeNode(Node<K, V>* node);
        Node<K, V>* getNode(K key);
        Node<K, V>* successor(Node<K, V>* node);
```

```
class TreeSet {
    private:
        TreeMap<E, bool>* map;

class Iterator {
    public:
    typename TreeMap<E, bool>::Iterator mItr;
```

Work Description - Testing Results - Script

```
#!/bin/bash

echo -n "Your program name: "
read program_name
echo -n "Begin Value: "
read beginValue
echo -n "End Value: "
read endValue

num=0;

for ((N=$beginValue; N<=$endValue; N=N*10))
do
        ((num++))
        echo "TEST CASE $num"
        echo "size: "$N
        echo -n "My Cpp STL: "
        (time -p ./$program_name $N) 2>&1 | grep real | sed 's/real/TIME/'
        echo -n "Offical Java STL: "
        (time -p java $program_name $N) 2>&1 | grep real | sed 's/real/TIME/'
        echo ""
done
```

Work Description - Testing Results - Script

```
#!/bin/bash

echo -n "Your program name: "
read program_name
echo -n "Begin Value: "
read beginValue
echo -n "End Value: "
read endValue

num=0;

for ((N=$beginValue; N<=$endValue; N=N*10))
do
        ((num++))
        echo "TEST CASE $num"
        echo "size: "$N
        echo -n "My Cpp STL: "
        (time -p ./$program_name $N) 2>&1 | grep real | sed 's/real/TIME/'
        echo -n "Offical Java STL: "
        (time -p java $program_name $N) 2>&1 | grep real | sed 's/real/TIME/'
        echo ""
done
```

Work Description - Testing Results - ArrayList&LinkedList

ArrayList

Test Case	Data Size	My Cpp STL	Official Java STL
1	10^5	0.06	0.86
2	10^6	0.03	0.10
3	10^7	0.30	1.12
4	10^8	3.21	23.80
5	10^9	29.07	39.04

LinkedList

Test Case	Data Size	My Cpp STL	Official Java STL
1	10^4	0.02	0.11
2	10^5	0.02	0.07
3	10^6	0.07	0.13
4	10^7	0.65	0.97
5	10^8	6.28	24.72

Work Description - Testing Results - ArrayList&LinkedList

ArrayList

Test Case	Data Size	My Cpp STL	Official Java STL
1	10^5	0.06	0.86
2	10^6	0.03	0.10
3	10^7	0.30	1.12
4	10^8	3.21	23.80
5	10^9	29.07	39.04

LinkedList

Test Case	Data Size	My Cpp STL	Official Java STL
1	10^4	0.02	0.11
2	10^5	0.02	0.07
3	10^6	0.07	0.13
4	10^7	0.65	0.97
5	10^8	6.28	24.72

Work Description - Testing Results - ArrayList&LinkedList

ArrayList

Test Case	Data Size	My Cpp STL	Official Java STL
1	10^5	0.06	0.86
2	10^6	0.03	0.10
3	10^7	0.30	1.12
4	10^8	3.21	23.80
5	10^9	29.07	39.04

LinkedList

Test Case	Data Size	My Cpp STL	Official Java STL
1	10^4	0.02	0.11
2	10^5	0.02	0.07
3	10^6	0.07	0.13
4	10^7	0.65	0.97
5	10^8	6.28	24.72

Work Description - Testing Results - HashSet&TreeSet

HashSet

Test Case	Data Size	My Cpp STL	Official Java STL
1	10^4	0.01	0.09
2	10^5	0.02	0.11
3	10^6	0.16	0.18
4	10^7	1.46	1.10
5	10^8	14.77	40.58

TreeSet

Test Case	Data Size	My Cpp STL	Official Java STL
1	10^3	0.00	0.09
2	10^4	0.01	0.09
3	10^5	0.09	0.12
4	10^6	1.25	0.92
5	10^7	18.92	18.99

Work Description - Testing Results - HashSet&TreeSet

HashSet

Test Case	Data Size	My Cpp STL	Official Java STL
1	10^4	0.01	0.09
2	10^5	0.02	0.11
3	10^6	0.16	0.18
4	10^7	1.46	1.10
5	10^8	14.77	40.58

TreeSet

Test Case	Data Size	My Cpp STL	Official Java STL
1	10^3	0.00	0.09
2	10^4	0.01	0.09
3	10^5	0.09	0.12
4	10^6	1.25	0.92
5	10^7	18.92	18.99

Work Description - Testing Results - HashSet&TreeSet

HashSet

Test Case	Data Size	My Cpp STL	Official Java STL
1	10^4	0.01	0.09
2	10^5	0.02	0.11
3	10^6	0.16	0.18
4	10^7	1.46	1.10
5	10^8	14.77	40.58

TreeSet

Test Case	Data Size	My Cpp STL	Official Java STL
1	10^3	0.00	0.09
2	10^4	0.01	0.09
3	10^5	0.09	0.12
4	10^6	1.25	0.92
5	10^7	18.92	18.99

Conclusion - Gains & Advices

- 提高了 C++ 的編譯能力，对 C++ 與 Java 的區別有了更感性的認識
- 加強了寫工程的能力
- 希望大作業可以提供更多的自由度

Conclusion - Gains & Advices

- 提高了 C++ 的編譯能力，对 C++ 與 Java 的區別有了更感性的認識
- 加強了寫工程的能力
- 希望大作業可以提供更多的自由度

Conclusion - Gains & Advices

- 提高了 C++ 的編譯能力，对 C++ 與 Java 的區別有了更感性的認識
- 加強了寫工程的能力
- 希望大作業可以提供更多的自由度

Conclusion - Gains & Advices

- 提高了 C++ 的編譯能力，对 C++ 與 Java 的區別有了更感性的認識
- 加強了寫工程的能力
- 希望大作業可以提供更多的自由度

Thank you for your attention