Week 7 – Laboratory Dictionaries

CSCI-2421 – Data Structures and Program Design
Javier Pastorino

1

Week 7 – Lab

- Introduction to Dictionaries
- Implementing `ArrayDictionary`
  - Uses dynamic memory allocation.
- Using the Dictionary and the BAG to implement (simulate) a Database and Database Join
- Chapter 18

OUTLINE | READINGS

CSCI2421 – Data Structures and Program Design J. Pastorino 2

2

The ADT Dictionary

- A collection of data about certain cities
- Consider need to search such a collection for
  - City
  - Country
- Criterion chosen for search is **search key**
- Dictionaries are a.k.a. **key-value stores**
- The ADT dictionary uses a search key to identify its entries

City	Country	Population
Buenos Aires	Argentina	13,639,000
Cairo	Egypt	17,816,000
Johannesburg	South Africa	7,618,000
London	England	8,586,000
Madrid	Spain	5,427,000
Mexico City	Mexico	19,463,000
Mumbai	India	16,910,000
New York City	U.S.A.	20,464,000
Paris	France	10,755,000
Sydney	Australia	3,785,000
Tokyo	Japan	37,126,000
Toronto	Canada	6,139,000

3

The ADT Dictionary
Specification

```
Dictionary
+isEmpty(): boolean
+getNumberOfEntries(): integer
+add(searchKey: KeyType, newValue: ValueType): boolean
+remove(targetKey: KeyType): boolean
+clear(): void
+getValue(targetKey: KeyType): ValueType
+contains(targetKey: KeyType): boolean
+traverse(visit(value: ValueType): void): void
```


4

```
6   template<class KeyType, class ItemType>
7   class DictionaryInterface {
8   public:
9       /**...*/
13      virtual bool isEmpty() const = 0;
14
15      /**...*/
19      virtual int getNumberOfItems() const = 0;
20
21      /**...*/
29      virtual bool add(const KeyType &searchKey, const ItemType &newItem) = 0;
30
31      /**...*/
37      virtual bool remove(const KeyType &searchKey) = 0;
38
39      /**...*/
42      virtual void clear() = 0;
43
44      /**...*/
51      virtual ItemType getItem(const KeyType &searchKey) const noexcept(false) = 0;
52
53      /**...*/
59      virtual bool contains(const KeyType &searchKey) const = 0;
60
61      /**...*/
66      virtual void traverse(void visit(ItemType &)) const = 0;
67
68      /**...*/
71      virtual ~DictionaryInterface() = default;
```

CSCI2421 – Data Structures and Program Design
J. Pastorino

ADT Dictionary Interface

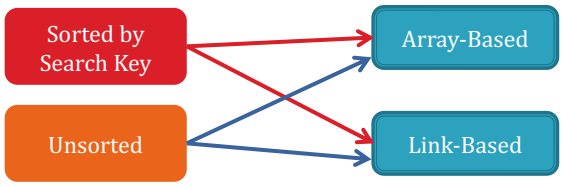
5

5

ADT Dictionary

Possible Implementations

- Categories of linear implementations



- A dictionary entry

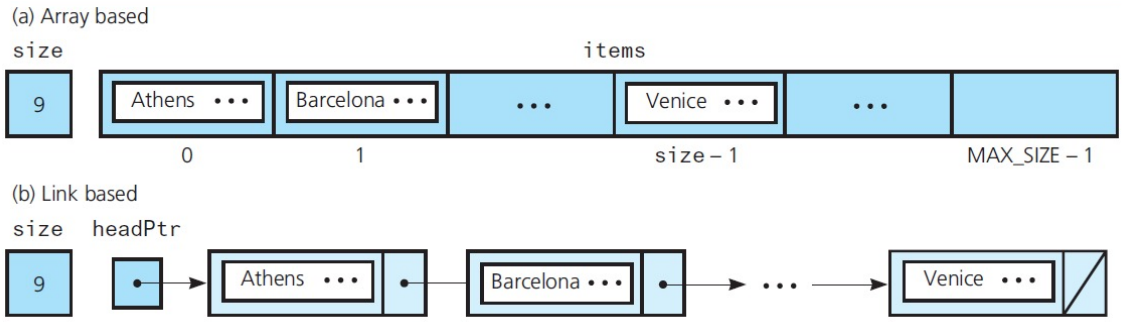
CSCI2421 – Data Structures and Program Design J. Pastorino 6

6

ADT Dictionary

Sorted Linear Implementations

- Data members for two sorted linear implementations



7

```
template<class KeyType, class ItemType>
class Entry {
private:
    KeyType searchKey;
    ItemType item;

protected:
    void setKey(const KeyType &aSearchKey) { searchKey = aSearchKey; }

public:
    explicit Entry() = default;

    explicit Entry(const KeyType &searchKey, const ItemType &newItem) : searchKey(searchKey), item(newItem) {};

    ItemType getItem() const { return item; }

    KeyType getKey() const { return searchKey; }

    void setItem(const ItemType &newItem) { item = newItem; }

    bool operator==(const Entry<KeyType, ItemType> &rightHandItem) const {
        return (searchKey == rightHandItem.getKey());
    }

    bool operator>(const Entry<KeyType, ItemType> &rightHandItem) const {
        return (searchKey > rightHandItem.getKey());
    }
```

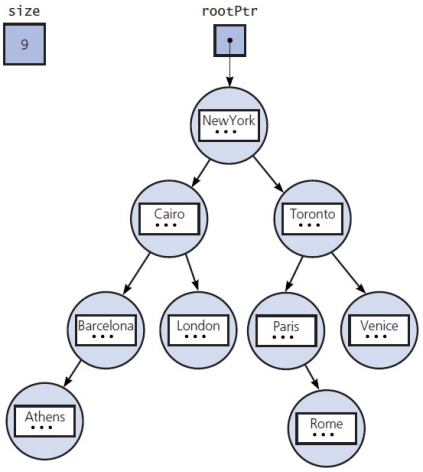
ADT Dictionary
Entry Class

8

ADT Dictionary

Binary Search Tree

- The data members for a binary search tree implementation



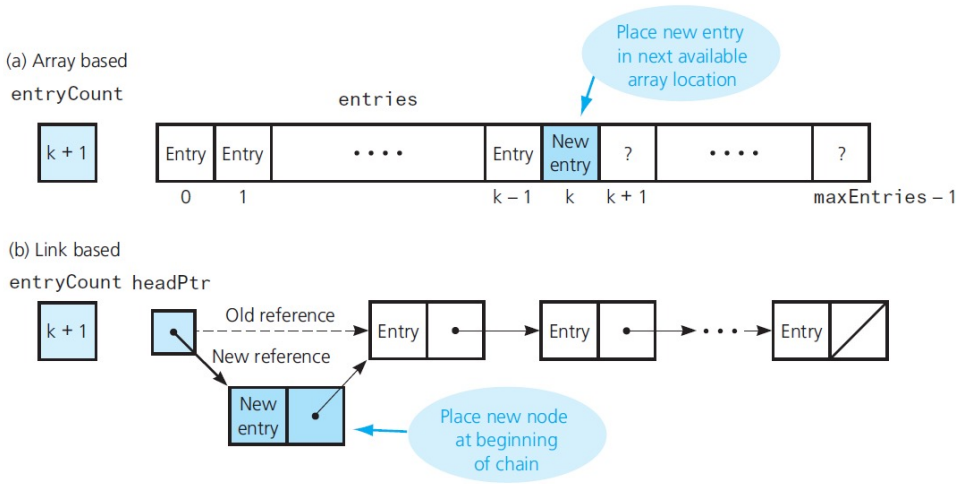
9

```
template<class KeyType, class ItemType>
class ArrayDictionary : public DictionaryInterface<KeyType, ItemType> {
private:
    static const int DEFAULT_CAPACITY = 50;
    unique_ptr<Entry<KeyType, ItemType>[]> items;      // Array of dictionary entries
    int itemCount;                                     // Current count of dictionary items
    int maxItems;                                      // Maximum capacity of the dictionary
    void destroyDictionary();

    int findEntryIndex(int firstIndex, int lastIndex, const KeyType &searchKey) const;

public:
    ArrayDictionary();

    explicit ArrayDictionary(int maxNumberOfEntries);

    ArrayDictionary(const ArrayDictionary<KeyType, ItemType> &dict);

    virtual ~ArrayDictionary();

    bool isEmpty() const;

    int getNumberOfItems() const;

    bool add(const KeyType &searchKey, const ItemType &newItem);

    bool remove(const KeyType &searchKey);

    void clear();

    ItemType getItem(const KeyType &searchKey) const noexcept(false);

    bool contains(const KeyType &searchKey) const;

    void traverse(void visit(ItemType &)) const;
};
```


ADT Dictionary Sorted Array-Based Implementation

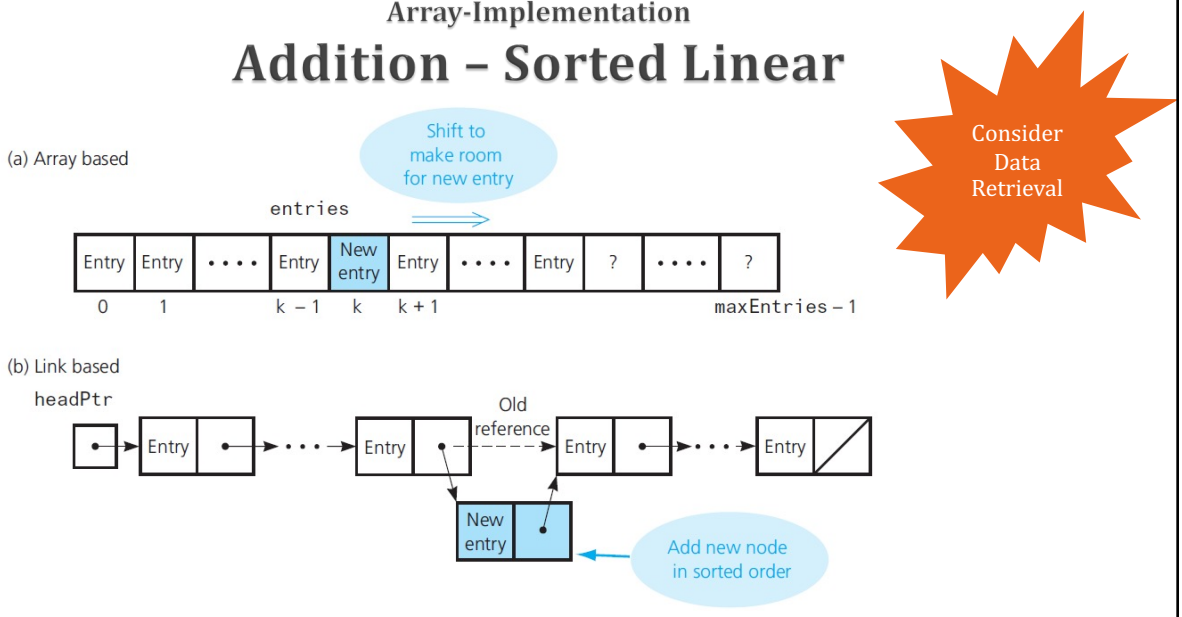
10

Array-Implementation
Addition – Unsorted Linear



11

Array-Implementation
Addition – Sorted Linear



12

Database Join

Cities Table

City	Country
Buenos Aires	Argentina
Cairo	Egypt
Johannesburg	South Africa
London	England
Madrid	Spain
Mexico City	Mexico
Mumbai	India
New York City	U.S.A.
Paris	France
Sydney	Australia
Tokyo	Japan
Toronto	Canada

Population Tables

City	Population
Buenos Aires	13,639,000
Cairo	17,816,000
Johannesburg	7,618,000
London	8,586,000
Madrid	5,427,000
Mexico City	19,463,000
Mumbai	16,910,000
New York City	20,464,000
Paris	10,755,000
Sydney	3,785,000
Tokyo	37,126,000
Toronto	6,139,000