DATA STRUCTURES & ALGORITHMS

DICTIONARIES AND HASHING

issntt@nus.edu.sg

Keys

A key is an **attribute** or a set of attributes whose **values uniquely** identify an **object**

Given a key, we can retrieve a **respective** whole object

User ID	Name	Gender	YOB
1	Albert Einstein	M	1879
2	Helen Keller	F	1880
3	Alfred Adler	M	1870
4	Mark Twain	M	1835
5	Maria Montessori	F	1870

What is the **unique** YOB of the user whose User ID is 2?

What is the **unique** User ID of the user whose YOB is 1870?

Problem

Write a method that, given a **list of Users** and a **User ID**, return the **respective User object**

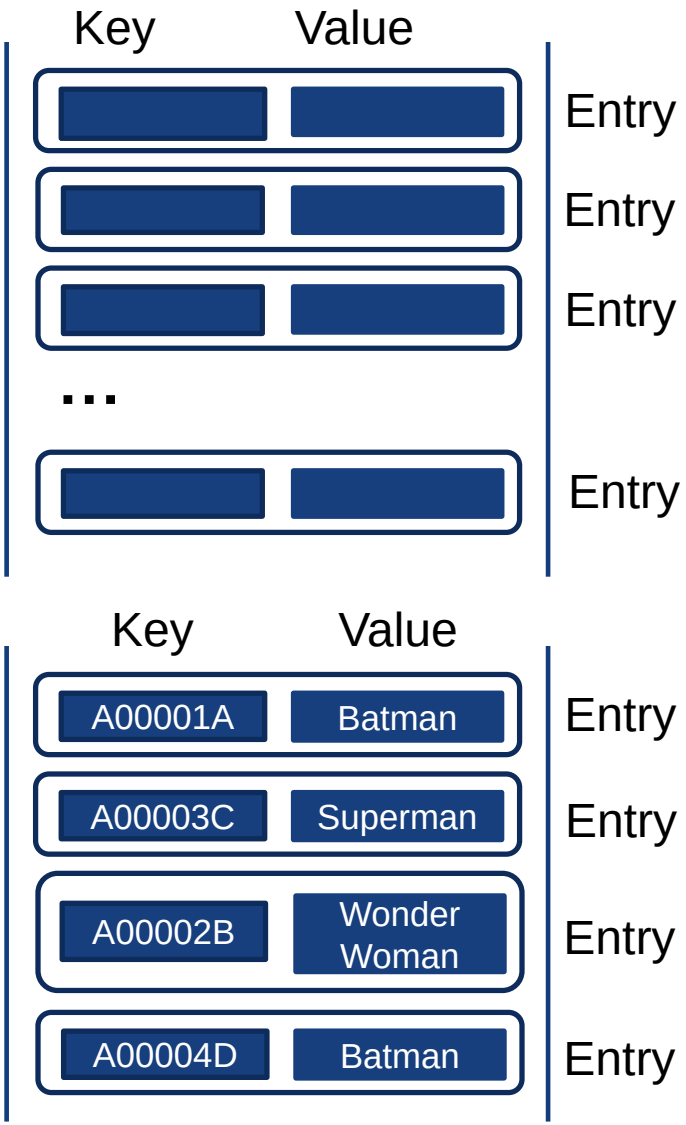
```
class User {
  public string UserId { set; get; }
  public string Name { set; get; }
  public string Gender { set; get; }
  public int YOB { set; get; }
  //...
}
```

Outline

- **Dictionary ADT**
- Using Dictionary ADT
- Implementing Dictionary ADT

Dictionaries

- A dictionary is a table containing a **set of key-value pairs**
- Both key and value are **objects**
- A key is **unique** while a value may **not**



Key Operations

- ***Add (key, value)***: add the pair (key, value) to the dictionary
- ***Get (key)***: given the key, retrieves the respective value from the dictionary

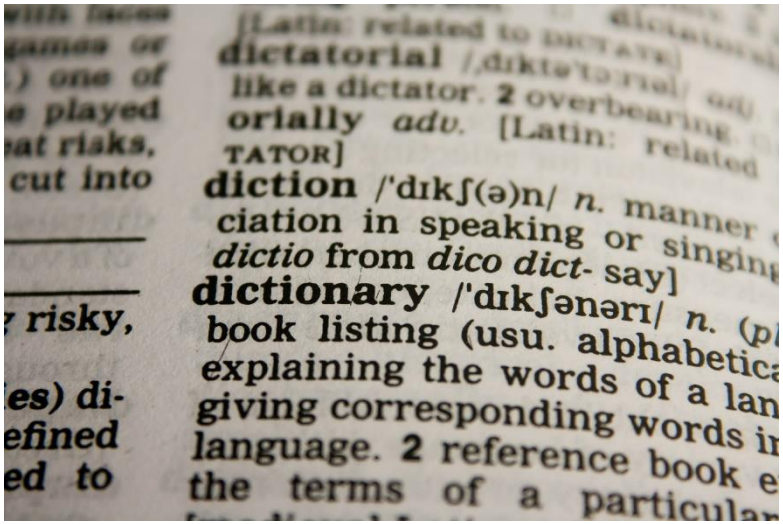


Image by PDPics from Pixabay

In fact, like Lists, Dictionaries can **add** a new entry. We can also **locate**, **retrieve** or **remove** an entry. The differences is Dictionaries **use keys**, **rather than positions** or the **values** as the input.

C# Generic Dictionary ADT

Method and Description
Add(TKey key, TValue value) Adds the specified key and value to the dictionary.
TryAdd(TKey, TValue) Attempts to add the specified key and value to the dictionary.
Item[TKey key] Gets or sets the value associated with the specified key.
TryGetValue(TKey, TValue) Gets the value associated with the specified key.
Keys Gets a collection containing the keys in the Dictionary<TKey,TValue>.
Values Gets a collection containing the values in the Dictionary<TKey,TValue>.
ContainsKey(TKey key) Determines whether the Dictionary<TKey,TValue> contains the specified key.
ContainsValue(TValue value) Determines whether the Dictionary<TKey,TValue> contains a specific value.
Count Gets the number of key/value pairs contained in the Dictionary<TKey,TValue>.
Remove(TKey) Removes the value with the specified key from the Dictionary<TKey,TValue>.
Clear() Removes all keys and values from the Dictionary<TKey,TValue>.

https://docs.microsoft.com/en-us/dotnet/api/system.collections.generic.dictionary-2

Study hint: these methods are listed in CRUD order

Outline

- Dictionary ADT
- **Using Dictionary ADT**
  - **Word Frequency**
- Implementing Dictionary ADT

Quiz What is the output?

```
public static void UsingDictionary()
{
  Dictionary<string, string> openWith =
                    new Dictionary<string, string>();
  openWith.Add("txt", "notepad.exe");
  openWith.Add("bmp", "paint.exe");
  openWith.Add("rtf", "wordpad.exe");

  Console.WriteLine(openWith["bmp"]);
  Console.WriteLine(openWith["rtf"]);

  openWith["rtf"] = "winword.exe";
  Console.WriteLine(openWith["rtf"]);

  if (openWith.ContainsKey("tif")) {
    Console.WriteLine("For key = \"tif\", value = {0}.",
                                          openWith["tif"]);
  }
  else {
    Console.WriteLine("Key = \"tif\" is not found.");
  }
}
```

Problem: Word Frequency

Given an **array of words**

Write a static method that displays **every word** and its respective number of **occurrences**

Input: [“good”, “morning”, “class”, “have”, “a”, “good”, “class”, “and”, “have”, “fun”]

Sample Output *(order of words is not important)*

```
good 2
morning 1
class 2
have 2
a 1
and 1
fun 1
```

Quiz Solution – Step 1

Loop through each **word**, and **fill in** a **dictionary**

```
public static Dictionary<string, int>
                    CountOccurs(string[] words) {
1 Dictionary<string, int> wordOccurDict =
                     new Dictionary<string, int>();
2 foreach (string word in words) {
  3 if (wordOccurDict.ContainsKey(word)) {
    4 int currrentOccurs = wordOccurDict[word];
      wordOccurDict[word] = currrentOccurs + 1;
    }
    else {
    5 wordOccurDict.Add(word, 1);
    }
  }

  return wordOccurDict;
}
```

Quiz Solution – Step 2

Loop through each **key**, display the key (word) and its value (occurrences)

```
static void DisplayOccurs(
        Dictionary<string, int> wordOccurDict) {
  Dictionary<string, int>.KeyCollection words =
      1 wordOccurDict.Keys;

2 foreach (string word in words) {
     Console.WriteLine("{0} {1}", word,
                       3 wordOccurDict[word]);
  }
}

static void CountAndDisplayOccurs(String[] words) {
  Dictionary<string, int> wordOccurDict = CountOccurs(words);
  DisplayOccurs(wordOccurDict);
}
```

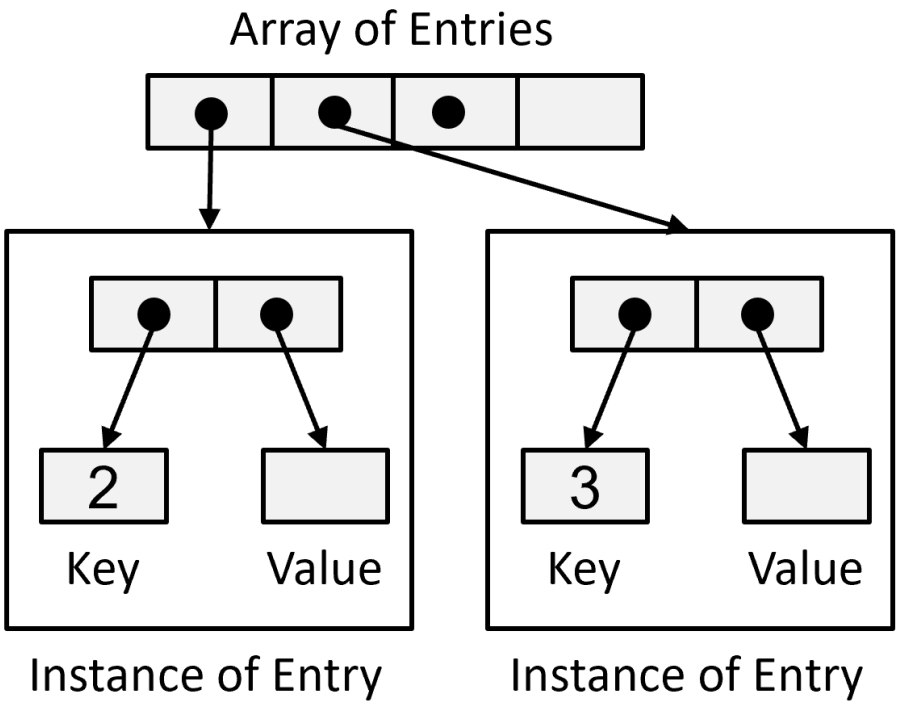
An **alterative** way is to **loop** through each **pair,** as in **here**

Outline

- Dictionary ADT
- Using Dictionary ADT
- **Implementing Dictionary ADT**
  - **Using Arrays**
  - Using Linked List (self exploration)

Using Arrays

Keep an **array of entries**, each of which encapsulate a **key** and **corresponding value**



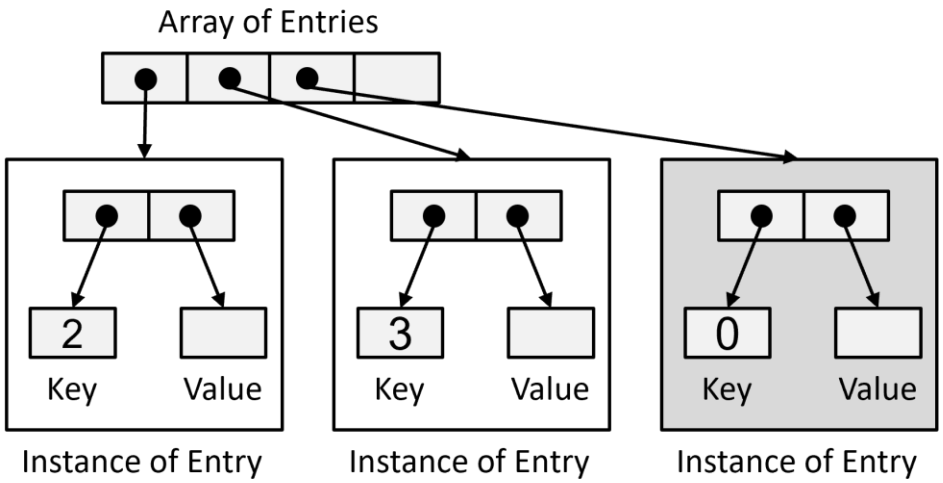

```
class Entry {
  public int Key
  {
    set; get;
  }
  public string Value
  {
    set; get;
  }
}
```

In this example, type of key is ***int*** and type of value is ***string***. Note that the type of either key or value can be anything. E.g., key is ***string*** and value is ***Student***

Implementing Operation Add

Algorithm for Add(key, value)
// Adds a new key-value entry to the dictionary. If key already exists,
// throws an ArgumentException

```
Search the array for an entry containing key
if (an entry containing key is found)
   Throw an ArgumentException
else
   Insert the new entry into the last position of the array
   Increment the size
```

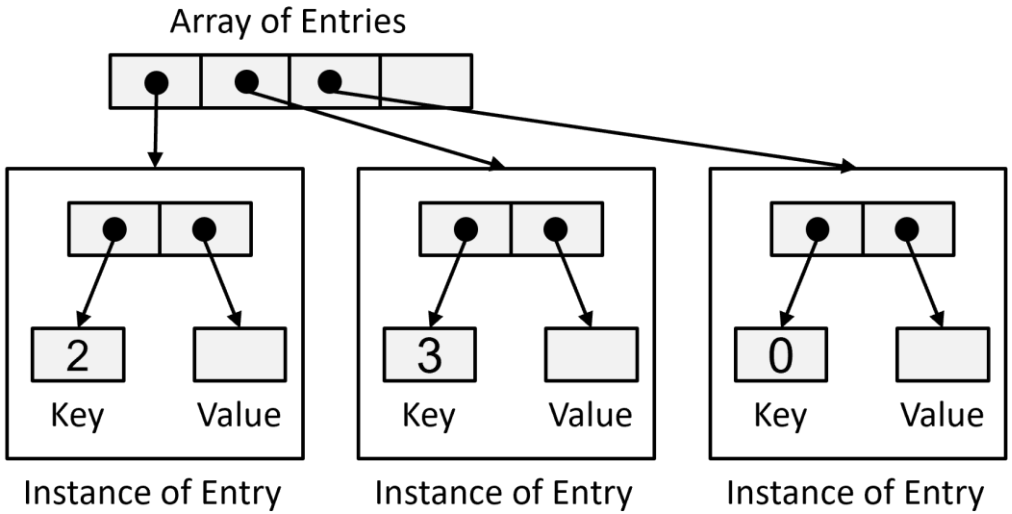


What is the running time of this algorithm?

Implementing Operation Get

Algorithm for Get(key)
// Retrieve the respective value for the given key. If key does not exist, return null

```
Search the array for an entry containing key
if (an entry containing key is found)
   Return the respective value
else
   Return null
```



What is the running time of this algorithm?

Implementing Lists with Arrays

Self study

Study by yourself the implementation of:

- Method *Remove(key)*
- Method *ContainKey(key)*
- Method *ContainValue(value)*
- Method *Keys()*
- Method *Values()*

Runtime Efficiency

The following table summarized the worst-case efficiencies of some dictionary operations

Operations	Array implementation	Linked List implementation
`Add`	O(n)	O(n)
`Get`	O(n)	O(n)
`Remove`	O(n)	O(n)
`Keys`	O(n)	O(n)

In many apps where get - retrieving the respective value from a key - is the primary operation. Is there any way to make it **faster**?

Readings

- Data structures and abstractions with Java, 4ed – Chapter 19, Dictionaries, *Frank M.Carrano and Timothy M. Henry*
- Data structures and abstractions with Java, 4ed – Chapter 20, Dictionary implementations, *Frank M.Carrano and Timothy M. Henry*