Computational thinking, problem-solving and programming:
Connecting computational thinking and program design

IB Computer Science

Content developed by
Dartford Grammar School
Computer Science Department
HL Topics 1-7, D1-4

1: System design
2: Computer Organisation
3: Networks
4: Computational thinking
5: Abstract data structures
6: Resource management
7: Control
D: OOP
HL & SL 4.2 Overview

4.2.1 Describe the characteristics of standard algorithms on linear arrays

4.2.2 Outline the standard operations of collections

4.2.3 Discuss an algorithm to solve a specific problem

4.2.4 Analyse an algorithm presented as a flow chart

4.2.5 Analyse an algorithm presented as pseudocode

4.2.6 Construct pseudocode to represent an algorithm

4.2.7 Suggest suitable algorithms to solve a specific problem

4.2.8 Deduce the efficiency of an algorithm in the context of its use

4.2.9 Determine the number of times a step in an algorithm will be performed for given input data
Topic 4.2.2

Outline the **standard operations of collections**
Collections?

- As far as the IB is concerned, collections are **UNORDERED** lists usually of **UNKNOWN length** or size.
- In practice we usually program collections using **LinkedLists** in Java.
- This means that we must remember there are a few things that LinkedLists **CAN** do that collections **CANNOT**.
Standard collection operations

- `.addItem( data )` = add data item to the collection
- `.resetNext()` = start at the beginning
- `.hasNext()` → tells whether there is another item in the list
- `.getNext()` → retrieves a data item from the collection
- `.isEmpty()` → check whether collection is empty

Be careful not to use methods not on this list, like `.size()` or `.length()`

ONLY these methods are allowed
# IB: Collection methods

## Collections

Collections store a set of elements. The elements may be of any type (numbers, objects, arrays, strings, etc.).

A collection provides a mechanism to iterate through all of the elements that it contains. The following code is guaranteed to retrieve each item in the collection exactly once.

```java
// STUFF is a collection that already exists
STUFF.resetNext();
loop while STUFF.hasNext()
    ITEM = STUFF.getNext();
    // process ITEM in whatever way is needed
end loop
```

<table>
<thead>
<tr>
<th>Method name</th>
<th>Brief description</th>
<th>Example: HOT, a collection of temperatures</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>addItem()</td>
<td>Add item</td>
<td>HOT.addItem(42) HOT.addItem(&quot;chile&quot;)</td>
<td>Adds an element that contains the argument, whether it is a value, string, object, etc.</td>
</tr>
<tr>
<td>getNext()</td>
<td>Get the next item</td>
<td>TEMP = HOT.getNext()</td>
<td>getNext() will return the first item in the collection when it is first called. Note: getNext() does not remove the item from the collection.</td>
</tr>
<tr>
<td>resetNext()</td>
<td>Go back to the start of the collection</td>
<td>HOT.resetNext() HOT.getNext()</td>
<td>Restarts the iteration through the collection. The two lines shown will retrieve the first item in the collection.</td>
</tr>
<tr>
<td>hasNext()</td>
<td>Test: has next item</td>
<td>if HOT.hasNext() then</td>
<td>Returns TRUE if there are one or more elements in the collection that have not been accessed by the present iteration: The next use of getNext() will return a valid element.</td>
</tr>
<tr>
<td>isEmpty()</td>
<td>Test: collection is empty</td>
<td>if HOT.isEmpty() then</td>
<td>Returns TRUE if the collection does not contain any elements.</td>
</tr>
</tbody>
</table>
Collections (Pseudocode)

NAMES = new Collection()
NAME = ""
loop while NAME <> "quit"
    input NAME
    if NAME <> "quit" then
        if NAMES.contains(NAME) then
            output NAME , " returned"
            NAMES.remove(NAME)
        else
            output NAME , " is leaving"
            NAMES.addItem(NAME)
        end if
    end if
end loop

output "The following students left and did not return"

NAMES.resetNext()
loop while NAMES.hasNext()
    output NAMES.getNext()
end loop

Task:
This program inputs NAMES of students who are leaving school early - for example to visit the doctor. The names are collected in a Collection list. When a student returns, tying the same name again removes that name from the list. At the end of the day, the secretary types "quit" to end the program and see a list of all students who left but did not return.