COL106 - Data Structures and Algorithms

Representation Independence

How can we optimize SimpleStr?

Hint:- Do we really need to copy the array in Substr?

Obs: Simple string is **Immutable**

$\Rightarrow$ array contents do not change

What if we want even more abstraction ?

→ Subclasses and class hierarchy

extend the functionality. without changing the "base class"

→ Interfaces (and hierarchy of interfaces)

Specify CONTRACTS of behavior (extend them hierarchically to form interface hierarchy)

Java: **Interfaces** and **Abstract Classes** for Behavior Abstraction

- Java enforces an application programming interface (API) through **interface**
  - An interface is a collection of method declarations with no data and no bodies.
- Interfaces do not have constructors, and they cannot be directly instantiated.
- A class **implements** an interface,
  - Then it must implement all the methods declared in the interface.
  - That is, it implements the expected behavior
- An **abstract class** also cannot be instantiated, but it can define one or more common methods that all implementations of the abstraction will have.

Subclass

class B extends A { } ····: (B.java)

Interface

```
enterface X { }        .... (X.java)
enterface Y extends X { }   . . . (Y.java)
```

Example from Java Standard Library

```
package java.lang;
public interface Comparable<T> {
        int compareTo (T o);
}
```

This interface imposes a total ordering on the objects of each class that implements it.

compareTo() - compares **this** object with the specified object for order. Returns a negative integer, zero, or a positive integer as this object is less than, equal to, or greater than the specified object.

from JDK documentation

```
package java.lang;
...
...
public final class Integer extends Number
implements Comparable <Integer>,... {
        ...
        public final int value;

        int compareTo (Integer o) {
                int x = this.value;
                int y = o.value;
                return (x < y) ? -1 : ((x==y) ? 0 : 1);
        }
        ...
}
```

ADTs for Collections

- Collection ADT

- A way to hold multiple objects

must offer.

(a) access objects in [illegible]ner

(b) storage

(c) iterate / traversal

Which should be exposed to users of collection?

Collection ADTs

a few basic ones

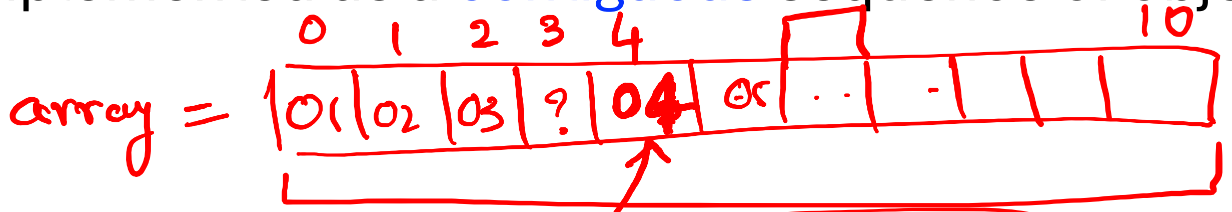
Array : offers access / update by index

List : offers sequential ordering and insert / delete

Set : ensures objects are unique (and no ordering guarantees)

Array – a Natural Way to Store Data

- Typically, implemented as a contiguous sequence of objects



array[4] ≡ 04

array[0] ≡ 01

array[10] ≡ error "Index Out of Bounds Exception"

not in Java ~~array[-1] ≡ last element~~

ARRAY

ADT

Given a set S, f(S) is the set of all functions from a finite set of non-negative integers to S.

Object

isEmpty()
readIndex(i)
insert(x, i)
delete(i)