Abstract Data Types

data object

set or collection of instances

integer = {0, +1, -1, +2, -2, +3, -3, …}

daysOfWeek = {S,M,T,W,Th,F,Sa}

Data Object

instances may or may not be related

myDataObject = {apple, chair, 2, 5.2, red, green, Jack}

Data Structure

Data object +

relationships that exist among instances
and elements that comprise an instance

Among instances of integer

$369 < 370$

$280 + 4 = 284$

Data Structure

Among elements that comprise an instance

369

3 is more significant than 6

3 is immediately to the left of 6

9 is immediately to the right of 6

Data Structure

The relationships are usually specified by specifying operations on one or more instances.

add, subtract, predecessor, multiply

Linear (or Ordered) Lists

instances are of the form

$(e_0, e_1, e_2, \ldots, e_{n-1})$

where e_i denotes a list element

$n >= 0$ is finite

list size is n

Linear Lists

$L = (e_0, e_1, e_2, e_3, \ldots, e_{n-1})$

relationships

e_0 is the zero'th (or front) element

e_{n-1} is the last element

e_i immediately precedes e_{i+1}

Linear List Examples/Instances

Students in MyClass =

(Jack, Jill, Abe, Henry, Mary, …, Judy)

Exams in MyClass =

(exam1, exam2, exam3)

Days of Week = (S, M, T, W, Th, F, Sa)

Months = (Jan, Feb, Mar, Apr, …, Nov, Dec)

Linear List Operations—Length()

determine number of elements in list

$L = (a,b,c,d,e)$

length = 5

Linear List Operations—Retrieve(theIndex)

retrieve element with given index

L = (a,b,c,d,e)

Retrieve(0) = *a*

Retrieve(2) = *c*

Retrieve(4) = *e*

Retrieve(-1) = error

Retrieve(9) = error

Linear List Operations—IndexOf(theElement)

determine the index of an element

L = (a,b,d,b,a)

IndexOf(d) = 2

IndexOf(a) = 0

IndexOf(z) = -1

Linear List Operations—Delete(theIndex)

delete and return element with given index

$L = (a,b,c,d,e,f,g)$

Delete(2) returns c

and L becomes (a,b,d,e,f,g)

index of $d,e,f,$ and g decrease by 1

Linear List Operations—Delete(theIndex)

delete and return element with given index

$L = (a,b,c,d,e,f,g)$

Delete(-1) => error

Delete(20) => error

Linear List Operations—Insert(theIndex, theElement)

insert an element so that the new element has a specified index

L = (a,b,c,d,e,f,g)

Insert(0,h) => *L = (h,a,b,c,d,e,f,g)*

index of *a,b,c,d,e,f,* and *g* increase by *1*

Linear List Operations—Insert(theIndex, theElement)

L = (a,b,c,d,e,f,g)

Insert(2,h) => *L = (a,b,h,c,d,e,f,g)*

index of *c,d,e,f,* and *g* increase by *1*

Insert(10,h) => error

Insert(-6,h) => error

Data Structure Specification

- Language independent
  - Abstract Data Type

Linear List Abstract Data Type

```
AbstractDataType LinearList
{
  instances
    ordered finite collections of zero or more elements
  operations
    IsEmpty(): return  true iff the list is empty, false otherwise
    Length():  return the list size (i.e., number of elements in the list)
    Retrieve(index): return the indexth element of the list
    IndexO f(x): return the index of the first occurrence of  x in
            the list, return -1 if x is not in the list
    Delete(index):  remove and return the indexth element,
        elements with higher index  have their index reduced by 1
    Insert(theIndex, x): insert x as the indexth element, elements
        with theIndex >= index have their index increased by 1
}
```