Ch 13: Introduction to Classes
Part 4

CS 2308
Fall 2011

Jill Seaman

Lecture 13

Example: Integer List

- We will implement a List data type that behaves like an array but has bounds checking on the subscripts
- Basic operations:
  - IntegerList (int size)
    creates a list with size elements, initialized to 0
  - setElement (int i, int y)
    performs assignment, stores y at list sub i
  - getElement (int i) [returns an int]
    returns the element at subscript i
  - ~IntegerList() - the destructor

IntegerList declaration

- IntegerList.h

```
// IntegerList specification

  class IntegerList
  {
    private:
      int *list;           //ptr to array
      int numElements;     //number of elems
      bool isValid(int);   //validates subscripts

    public:
      IntegerList(int);          //constructor
      ~IntegerList();            //destructor
      void setElement(int,int); //element assignment
      int getElement(int);       //element access
  };
```


IntegerList Implementation

- IntegerList.cpp

```
// IntegerList implementation
#include <iostream>
using namespace std;

#include "IntegerList.h"

IntegerList::IntegerList(int size) {
   list = new int[size];
   for (int i = 0; i < size; i++)
      list[i] = 0;
   numElements = size;
}
IntegerList::~IntegerList() {
   delete [] list;
}

bool IntegerList::isValid(int element)
{
   return  !(element < 0 || element >= numElements);
}
```

IntegerList Implementation

- IntegerList.cpp cont.:

```
void IntegerList::setElement(int index, int value) {
   if (isValid(index))
      list[index] = value;
   else {
      //error throw exception
      cout << "Error: Invalid subscript" << endl;
      exit(EXIT_FAILURE);
   }
}

int IntegerList::getElement(int index) {
   if (isValid(index))
      return list[index];
   else {
      //error throw exception
      cout << "Error: Invalid subscript" << endl;
      exit(EXIT_FAILURE);
   }

}
```


A driver progam to demo IntegerList

- driver.cpp:

```
//using IntegerList class
#include<iostream>
#include "IntegerList.h"
using namespace std;
int main() {
   const int SIZE = 20;
   IntegerList numbers(SIZE);

   for (int x = 0;  x < SIZE; x++) {
      numbers.setElement(x,9);
      cout << "* ";       //output * for success
   }
   cout << endl;

   for (int x = 0; x < SIZE; x++)  //output elems
      cout << numbers.getElement(x) << " ";
   cout << endl;

   numbers.setElement(50,9); //elem out of bounds

   cout << "setElement 50 was successful" << endl;
   return 0;
}
```

Result of demo:

- execute:

```
* * * * * * * * * * * * * * * * * * * * *
9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
Error: Invalid subscript
```

- Note: did not output “setElement 50 was successful”

Things to note

- Details of how list was represented (as a dynamically allocated array) are hidden from the public interface.
- We could use a different representation for the list (a really big static array, linked list), and rewrite the implementation file, but not the driver file.
- The IntegerList is an example of an Abstract Data Type