CSPP 511-01: 
Introduction to Object-Oriented Programming

Harri Hakula

Ryerson 256, tel. 773-702-8584
hhakula@cs.uchicago.edu

July 12, 2000
Outline

- Abstract Data Types
- Data Structures
- Stack
- Mid-Term
Abstract Data Types

An Abstract Data Type (ADT) is a data type defined so that the programmers who use the type do not have to study how the values and operations are implemented.

The SS#-example from the last lecture is an ADT.

One of the advantages of the object-oriented programming is that creation of ADTs is natural.
Data Structures

Data structures are ways to organize data so that it is possible to operate on the data in some specific and consistent way.

Is an array a data structure or not? Well,

- data is organized as one contiguous memory location,
- every element can be accessed in the same way,
- every element can be accessed all the time.

We conclude that an array is a data structure which does not restrict the access to its elements.
Stack

A Stack is a data structure which only allows the user to access the last stored element. This is often called the LIFO (Last-In-First-Out). The elements are pushed to the stack and popped off the stack. The API needs to address these operations.

Let us consider a stack which is based on an array. In other words, elements are stored in an array but only the last one inserted, can be accessed at any moment. Once the last one in has been accessed, it is no longer on the stack, it has been popped.
Stack Cntd.

Note, that since an array must have a fixed size, this size is also the maximum capacity of the stack.

We can identify two special situations:

- One cannot push elements to a stack which is full.
- One cannot pop elements from an empty stack.

How do we set the capacity? It must be done before the stack is used.
Constructors

Constructor is a special method which is called exactly at the point when the object is instantiated. In JAVA that happens when the object is created with the operator new,

MyObject m = new MyObject();

Constructors can take arguments, the one without any is called the default constructor. The system will automatically provide one for every class. Automatic default constructor is not available if any other constructor is specified.

For our stack class, we might want to set the capacity from within the program, MyStack m = new MyStack(100), where MyStack would now hold 100 elements.
Stack: Design

We have not yet specified what kind of elements our Stack can hold. Let us consider a stack on ints for simplicity.

Public API:

`IntStack()` Constructor. We use fixed size for all IntStacks, therefore only the default constructor is given.

`void push(int)` Push the int on the stack.

`int pop()` Return the element on top of the stack and remove it.

`boolean isEmpty()` This is just to save us from popping from an empty stack.
Stack: Implementation

Implementation is straightforward: we need an array and a variable to indicate the top of the stack at any moment. Here is one way to code it:

```java
public class IntStack {
    private static final int MAX_CAPACITY = 100;
    public IntStack() {
        store_ = new int[MAX_CAPACITY];
        top_ = 0;
    }
    public void push(int value) {
        if (top_ == MAX_CAPACITY) {
            System.out.println(
```
```java
            System.out.println("Stack is full.");
            return;
        }
        store_[top_] = value;
        top_++;
    }
    
    public void pop() {
        System.out.println("Popped: ");
    }

    public int peek() {
        return store_[top_];
    }

    private int store_;
    private int top_;
```
"Error: stack is full, cannot push.";

return;

}

store_[top_++] = value;

}

public int pop() {

if(top_ == 0)
{

System.out.println(
    "Error: stack is empty, cannot pop.";

    return 0;

}

return store_[--top_];

}
public boolean isEmpty() {
    return (top_ == 0);
}

private int top_;  
private int[] store_;  

We shall see some interesting uses of stack next week. Also, does it not bother you that the capacity is fixed? Why cannot we have stacks of unlimited size?
Mid-Term

Mid-Term covers the material discussed on the first three weeks. Books are not allowed, bring a pen or pencil.

There are six questions with varying difficulty. You will have 45 minutes after which I shall present the correct answers.

Typical questions are:

- After the following assignments, what are the values of the variables?

- What does this loop print?

- Given this for-statement, write an equivalent while-loop.