Final Exam Review

Tom Kelliher, CS 116

Dec. 9, 2002

1 Administrivia

Announcements

Lab and postlab 7 due today.

Extra credit applet due Wednesday.

Final Friday in HS 153 3:00 to 5:00.

2 Example

This example pulls together a lot of what we've done this semester.

1. A stack is a fundamental data structure in CS.
2. Last in, first out (LIFO) principle: like a stack of trays in a cafeteria line.
3. Stack parameter: number of items stack can hold.
4. Stack state: data within the stack (array), top of stack.
5. Stack operations: empty, full, push, pop.

Integer stack implementation:

```
import java.applet.*;

public class Driver extends Applet
{
   Stack s;

   public void init()
   {
      int tempI;
      boolean tempB;

      s = new Stack(10);
      tempB = s.empty();   // Returns true.
      s.push(12);
      tempB = s.empty();   // Returns false.
      s.push(9);
      tempI = s.pop();     // Returns 9.
      tempI = s.pop();     // Returns 12.
      tempB = s.empty();   // Returns true.
   }
}

class Stack
{
   int topIndex;
   int data[];

   public Stack(int length)
   {
      topIndex = -1;
      data = new int[length];
   }

   boolean empty()
   {
      if (topIndex == -1)
         return true;
      else
         return false;
   }

   boolean full()
   {
      if (topIndex == data.length - 1)
         return true;
```

```
      else
         return false;
   }

   int pop()
   {
      if (empty())
      {
         // Throw an exception and return.
         return 0;
      }
      else
      {
         topIndex--;
         return data[topIndex + 1];
      }
   }

   void push(int newTop)
   {
      if (full())
      {
         // Throw an exception and return.
      }
      else
      {
         topIndex++;
         data[topIndex] = newTop;
      }
   }
}
```

3 Control and Calculation

1. Iterative execution: for, while, and do/while loops.

   When to use?

   Index variable.

2. Conditional execution: if/else if/else, switch, break.

   Purpose of the break within a switch case.

3. Conditions — keys to the above. Precedence, associativity, operators.

   Remember the precedence table.

4. Arithmetic: Operators and their types. Type conversion: implicit and explicit casts.

4 Methods

1. Parameter passing: Formal and actual parameters.

2. Pass by reference vs. pass by value.

3. Local variables.

4. Return values and the type of a method.

5. Special applet methods: init, paint, interface methods for listeners.

6. Special class methods: The constructor.

5 Data

1. Primitive types: int, double, boolean, char, etc.

2. Arrays: Type, number of elements, declaring, indexing, .length.

3. Objects: String, Label, TextField, etc.

   Creating your own.

4. Declaring and instantiating. The actual variable or a reference?

5. Variable/Object scope. Naming clashes:

```
double average(int data[])
{
   double average;   // dohhh....
```

6 Object-Oriented Design

1. What is the model?

2. Which object, which action?

3. extends, inheritance, sub- and super-classes. Overrides.

4. Public vs. private. Why use?

5. The use of instance variables

   The use of the constructor.

7 Event-Driven Programming

1. Types of events.

2. Listener classes must implement all interface methods. Interface methods must have code to deal with each event from each object.

3. Objects must register their events and designate handler objects.