

Week 3, Lecture 1

Coding Algorithms



1

Announcements

- **1st Assignment** is due at the end of next week (deadline **4pm Friday August 20th**).
- to submit your assignment go to the course home page, click on **Assignment Submission** and follow the instructions

2

Assignment 1 Submission

- prepare the file **Expanding Circles.java**
- complete the comments at the top of this file, e.g.:

```
// Family name: Xmas  
// Given names: Mary Jane  
// Student number: 12345678
```
- follow the instructions for submission

3

This Week

Lecture 1: Coding algorithms

Lecture 2: Control constructs

Java Genesis:

–**Ch4: Control constructs**

Lab Assessment 3 (deadline Week 5)

Quick Quiz for Ch3

4

Code Creation

1. **Specify** the problem
2. **Develop** a scenario to solve the problem
3. **Refine** the scenario into Java code
4. **Review** the code and modify/improve
5. **Test** the code for errors
6. **Verify** that the code works

5



Problem
finding the maximum

Problem Statement

Numbers are supplied one after the other. Find the maximum of this collection of numbers.

6

Specification

Find a number which

- is one of the numbers in the given collection of supplied numbers
- has the property that no other number in the collection is bigger

7

A Scenario

1. To begin, let the maximum-so-far be the first number supplied.
2. When a new number is supplied, let the maximum-so-far become the larger of the new number supplied and the previous value of the maximum-so-far.

8

```
import genesis.*;
public class Maximum {

    public static void main (String [ ] args) {
        int maxSoFar =
            DialogBox.requestInt("First number: ");
        Transcript.print(maxSoFar+" ");
        int nextNum;
        for (int i=2; i<=6; i++) {
            nextNum =
                DialogBox.requestInt("Next number: ");
            Transcript.print(nextNum+" ");
            maxSoFar = Math.max(maxSoFar, nextNum);
        }
        Transcript.println( );
        Transcript.println("maximum is "+maxSoFar);
    }
}
```

9

Verification

How can we verify that the algorithm works?

10

An Invariant

The value of the variable *maxSoFar* is always the maximum of the integers so far supplied.

Reason by structural induction:

- it is true initially (when the 1st number is supplied)
- it remains true after the for-loop block is executed each time

11

Observation

When trying to understand how a system changes, it is often helpful to concentrate on those aspects of the system that do not change

i.e. the **invariants!**

e.g. conservation of mass, energy, momentum

12



Problem calculating the gcd

Specification

Given two positive integers, find that number which is

- a common factor of both the given integers
- larger than any other common factor

13

A Scenario

1. Find the factors of the first integer.
2. Find the factors of the second integer.
3. List all the numbers that are factors of both of the given integers.
4. From this list select the largest integer.

14

Example

- factors of **36** are 1, 2, 3, 4, 6, 9, 12, 18, 36
- factors of **24** are 1, 2, 3, 4, 6, 12, 24
- common factors are 1, 2, 3, 4, 6, 12
- **12** is the largest integer in this list;
hence the $\text{gcd}(36, 24) = 12$

15

Euclid's Algorithm

1. Replace the larger of the two integers by the difference between the larger and the smaller.
2. Continually repeat this process until the two numbers are the same.
3. This common number is the gcd.

16

Example

- start with **36** and **24**
- replace **36** by **36-24** to give **12** and **24**
- replace **24** by **24-12** to give **12** and **12**
- hence the $\text{gcd}(36, 24) = 12$

17

```
import genesis.*;
public class Euclid {

    public static void main (String [ ] args) {
        int num1 =
            DialogBox.requestInt("first integer:");
        int num2 =
            DialogBox.requestInt("second integer:");
        Transcript.print
            ("gcd of "+num1+" and "+num2+" is ");
        while (num1 != num2) {
            if (num1 > num2) num1 = num1 - num2;
            else num2 = num2 - num1;
        }
        Transcript.println(num1);
    }
}
```

18

Verification

Can we prove that Euclid's algorithm works?

An Observation

If we have a pair of positive integers and replace the larger by the difference between it and the smaller, the pair of integers that result have the same common factors!

19

An Invariant

The gcd of *num1* and *num2* is unchanged and is the value we want to compute.

Reason by structural induction:

- It is true initially (when *num1* and *num2* are the integers supplied).
- It remains true after replacing the larger by the difference between it and the smaller.

20

Termination

Can we be sure Euclid's algorithm terminates?

21

An Observation

- The numbers *num1* and *num2* are always positive integers.
- At the completion of each execution of the while-loop, the maximum of *num1* and *num2* has strictly decreased.

22