Lecture VI:

Recursion, Data Structures + Sorting

Mark Huiskes

LIACS, Leiden University

Introduction to Programming, Media Technology MSc

Lecture Schedule

15 Sep	Lecture I	Variables, data types, operators + assignments
Today	Lecture II	Functions + mouse interaction
29 Sep		no class
6 Oct	Lecture III	Conditions + loops
13 Oct	Lecture IV	Arrays
20 Oct	Lecture V	Classes + designing complex programs
27 Oct	Lecture VI	Recursion, data structures + sorting
3 Nov		no class
10 Nov	Lecture VII	Images and libraries
17 Nov	Lecture VIII	Max/MSP/Jitter (Edwin van der Heide)

Today

1. Tips + tricks
2. Summary sketch
3. Recursion
4. Stacks and Queues
5. Sorting

Tips & Tricks

Sorting out Sorting - video:

http://video.google.com/videoplay?docid=3970523862559774879

'Documentary' made at the University of Toronto in 1980 by Ronald Baecker

Class versus Object

- A class defines a data type

  A data type characterizes how a set of entities is internally represented and which operations can be performed on it

  A class if often described as a blueprint or template for creating objects. A class defines the (abstract) characteristics that objects share: the properties and the methods.

- Objects are the actual data variables that have the class as data type

  Objects are instances of a class. All objects created from one class have the same property names and methods, but the variables of each can contain different data

Class definition – general form

```
class Class-name {
    property-declarations

    // constructor(s)
    Class-name ( arguments ) {
            constructor-statements
    }

    method-definitions
}
```

Declaration and allocation of objects

- Declaration: like for built-in types

  Class-name variable-name(s);

  Examples:
  ```
  Employee e1;
  Employee  e2, e3, randomGuy;
  ```

- Allocation: reserving memory for the object

  variable-name = new *constructor-call;*

  Examples:
  ```
  e1 = new Employee("John", 58);
  e2 = new Employee();
  ```

- Or both at once: `Employee e1 = new Employee("John", 58);`

Using Objects: dot notation

- Accessing properties
  object-variable.property

  Examples: println(e1.name);
  float salaryMay = e1.salary + holidayMoney;
  e1.department = "R&D";

- Calling methods
  object-variable.method-name(*arguments*)

Examples: e1.report(5);
randomGuy.newSalary(100000);
randomGuy.runAroundInBuilding();

Summary Sketch

Recursion

Jargon file:

Recursion: see Recursion.

Example of a normal function call

```
void setup() {
  float x = HALF_PI;
  float f = doubleIt(x);
}

float doubleIt(float x) {
  return(2 * x);
}
```

Example of a recursive function call

```
int factorial(int n) {
  if (n <= 1)
    return 1;
  else
    return (n * factorial(n - 1));
}
```

Recursion

- Recursive function: a function that calls itself

- Don't recursive functions lead to an infinite sequence of function calls?

  You have to make sure they don't!

- Recursion is a method of solving a problem by reducing it to a simpler problem of the same type

- Recursive definition: A definition in which something is defined in terms of smaller/simpler versions of itself

  Example: n! = n * (n-1)!

Analyzing recursion

- Example:

```
int factorial(int n) {
    if (n <= 1)
        return 1;
    else
        return (n * factorial(n - 1));
}
```

Example VI.1

Common Pattern

Make a problem-solving function:

- Make some (well-chosen) step(s) toward the solution
- You are left with a simpler problem similar to the original problem: call the problem-solving function again with arguments for the smaller problem
- When the problem is small enough to provide the answer directly, without computation, you return this answer (instead of calling the problem-solving function again)

String

- (special) Class for string variables
- Allows

  ```
  String s1 = "bike";
  ```

  to construct a string object
- Methods
  - length() : returns length of the string
  - charAt(int i): returns character at position i

    first character in string: `s1.charAt(0);`

    last character in string: `s1.charAt(s1.length() - 1)`
  - substring(int i1, int i2): returns substring consisting of characters from position i1 until (but not including!!) position i2

    Example:

    `s1.substring(1, 3);` returns "ik" (the second and third character of "bike")

Example VI.2

Example VI.3

Summing up

- Recursion can give very short and elegant solutions to programming problems
- Also useful for visual effects
- But they can always be solved without recursion as well

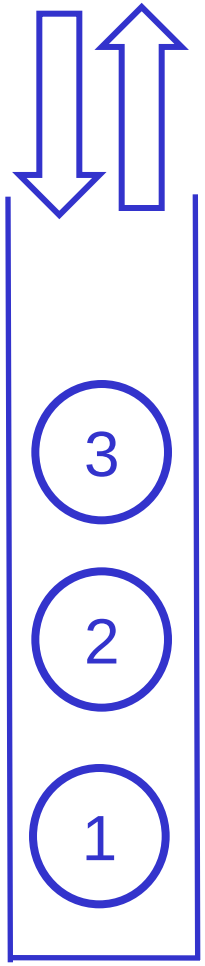
Data Structures

Useful Data Types

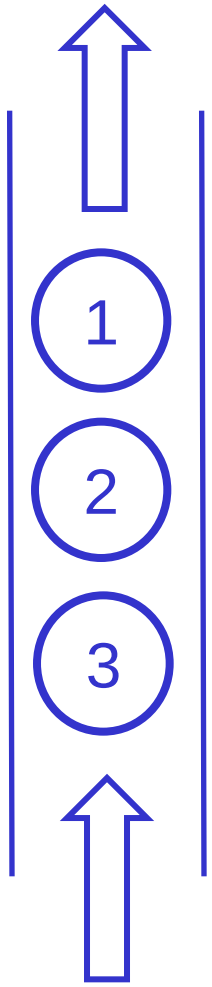
- Stack: push, pop
- Queue: enqueue, dequeue

- String
- Set
- (Linked) List
- Tree
- Graph

Often built on top of arrays (and references/pointers)



Stack: LIFO



Queue: FIFO

Stack

- A linear data structure with LAST IN, FIRST OUT (LIFO) access on one side

- Supports two main operations:
  - Push: put on top of the stack
  - Pop: take from the top of the stack

Example VI.4

Queue

- A linear data structure with FIRST IN, FIRST OUT (FIFO) access: adding elements on one side, and removing elements from the other side
- Application: waiting lines
- Supports two main operations:
  - Enqueue: put at rear of the queue
  - Dequeue: take from front of the queue

Example VI.5

Sorting

Sorting

- Putting a sequence of elements into ascending or descending order

- Some types of methods:
  - insertion methods
  - exchange methods
  - selection methods

Analyzing sorting algorithms

- Performance of sorting methods depends on circumstances: characteristics of data and hardware

- Performance is typically analyzed relative to the number of elements (n) of the sequence.
  - Computational complexity: growth of required number of movements and comparisons
  - Memory usage: growth of required memory

- Various other properties are of interest, e.g. stability

Example VI.6

first round

(8 2 5 100 1)

(2 8 5 100 1)

(2 5 8 100 1)

(2 5 8 1 100)

second round

(2 5 1 8 100)

third round

(2 1 5 8 100)

fourth round

(1 2 5 8 100)

Note: large items move to the end quickly; small ones bubble up slowly

More sorting

- Sorting out Sorting: see Tips + Tricks
- The Art of Computer Programming, Volume 3
- performance comparison: http://en.wikipedia.org/wiki/Sorting_algorithm

  check out Bozo sort: even worse than bubble sort…