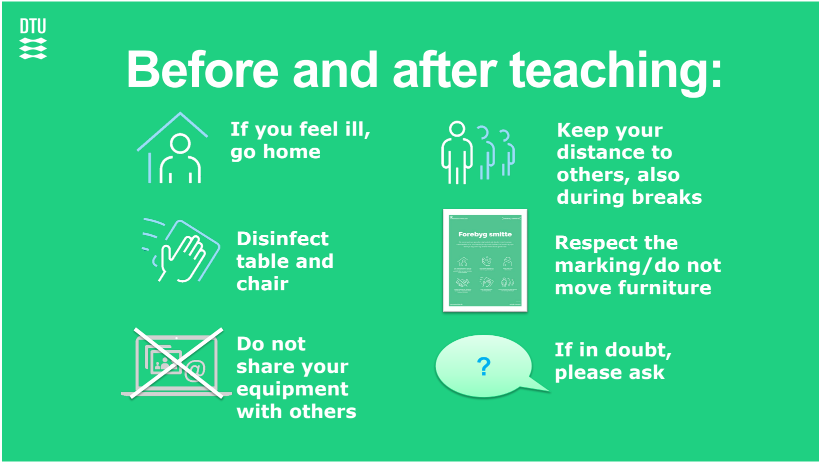
02393 Programming in C++



DTU

Before and after teaching:

If you feel ill, go home

Disinfect table and chair

Do not share your equipment with others

Keep your distance to others, also during breaks

Forebyg smitte

Respect the marking/do not move furniture

?

If in doubt, please ask

02393 Programming in C++
Module 11: Linked Lists

Lecturer:
Alceste Scalas

(Slides based on previous versions by Andrea Vandin, Alberto Lluch Lafuente, Sebastian Mödersheim)

17 November 2020

Lecture Plan

#	Date	Topic	Book chapter *
1	01.09	Introduction	
2	08.09	Basic C++	1
3	15.09	Data Types	2
4	22.09	Libraries and Interfaces	3
5	29.09		
6	06.10	Classes and Objects	4.1, 4.2 and 9.1, 9.2
		Autumn break	
7	20.10	Templates	4.1, 11.1
8	27.10	LAB DAY	Old exams
9	03.11	Inheritance	14.3, 14.4, 14.5
10	10.11	Recursive Programming	5
11	17.11	Linked Lists	10.5
12	24.11	Trees	13
13	01.12	Summary & Exam Preparation	
	07.12	Exam	

* Recall that the book uses sometimes ad-hoc libraries that are slightly different with respect to the standard libraries (e.g., strings and vectors).

Recursive Data Types

Many mathematical entities can be recursively defined:

- a natural number is 0, or the successor of a natural number
- a set can be empty, a singleton, or the union of two sets

Similarly, many data types can be recursively defined:

- a list can be empty, one item, or the concatenation of two lists
- a tree can be empty, one leaf node, or an internal node with two sub-trees
- ...

Linked Lists

Recursive Definition

```
struct Node {
    int content;
    Node *next;
}
```

A Node has some content and points to a Node

A list can be then just a pointer to a Node (i.e., *Node)

- A nullptr represents an empty list
- Otherwise, the pointer points to the first node of the list

Important: do not forget to initialize your pointers!

Linked Lists

Live Programming + Examples

In previous lectures we saw an example of how to implement a...

- vector class based on arrays

This week we are going to see examples on how to implement a...

- vector class based on linked lists
- set class based on linked lists

Note: linked lists are provided by the STL. See:
http://en.cppreference.com/w/cpp/container/list

Array vs. Lists

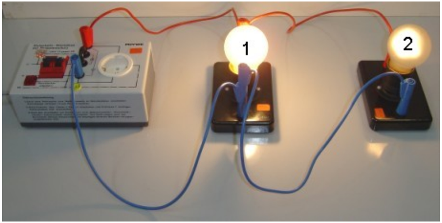
	Array	*List*
Iterative Access	$O(1)$	$O(1)$
Random Access	$O(1)$	$O(n)$
Insert/Delete	$O(n)$	$O(1)$[1]
Insert/Delete at end	$O(1)$[2]	$O(n)$

[1] given pointer to node before insertion/deletion

[2] amortized

Doubly-linked Lists

Some annoyances of single-linked lists: delete a pointed element, concatenate two lists, etc.
One possible solution: doubly-linked lists



Implementation	*Insert head*	*Concat*	*Reverse*
Concatenation by connecting the tail of one list with head of other list.	$O(1)$	$O(1)$	$O(N)$

Doubly-Linked Lists

Recursive Definition

```
struct Node {
    int content;
    Node *prev;
    Node *next;
}
```

A Node has some content and points to two Nodes: the previous one and the next one in the list

02393 Programming in C++

Before and after teaching:

If you feel ill, go home

Keep your distance to others, also during breaks

Disinfect table and chair

Respect the marking/do not move furniture

Do not share your equipment with others

If in doubt, please ask