# Algorithms and Data Structures 2

## Recap Lectures 9-12

**Dr Michele Sevegnani**

School of Computing Science
University of Glasgow

*michele.sevegnani@glasgow.ac.uk*

# Topics we covered so far

- **Non-comparison sorts**
  - COUNTING-SORT
  - RADIX-SORT
- **Linked lists**
  - Singly linked lists
  - Doubly linked lists
  - Circular doubly linked list with a sentinel
- **Abstract Data Types**
  - Stack
  - Queue
  - List

# Question 1

- **Briefly describe:**
  - The five the main operations of the List ADT
  - An iterative and a recursive definition of operation GET(L,i) in the implementation of the List ADT based on linked list.

# Question 1: solution

- **Main list operations**
  - GET(L,i): return the element at index i in list L, without removing it
  - SET(L,i,x): replace element at index i in list L with x and return previous element at index i
  - ADD(L,x): insert element x to the end of list L
  - ADD-AT(L,i,x): insert element x at index i in list L, shifting all elements after this
  - REMOVE(L,i): remove and return element at index i in list L, shifting all elements after this

# Question 1: solution (cont.)

```
GET(L,i)
  counter := 0
  x := L.head
  while x != NIL
    if counter = i
      return x
    counter := counter + 1
    x := x.next
  return x
```

Call with REC-GET(L.head,i)

```
REC-GET(x,i)
  if x = NIL or i = 0
    return x
  return REC-GET(x.next, i - 1)
```

# Question 2

- **Describe a recursive algorithm to determine if a string has more vowels than consonants**
  - Hint: write a recursive algorithm to return an integer representing number of vowels minus number of consonants

# Question 2: solution

- **Explain with pseudocode**
  - s is a string
  - Output: integer denoting number of vowels in s minus number of consonants in s

```
DIFF(s)
  if LENGTH(s) = 0            // empty string
    return 0
  if s = vs' and v is vowel   // v is s[0]
    return 1 + DIFF(s')
  else
    return -1 + DIFF(s')
```

# Question 2: solution (cont.)

- **We write explicitly the recursion trace for DIFF(aabcdee) (vowels in green):**

DIFF(aabcdee)

1 + DIFF(abcdee)

1 + 1 + DIFF(bcdee)

1 + 1 -1 + DIFF(cdee)

1 + 1 -1 -1 + DIFF(dee)

1 + 1 -1 -1 -1 + DIFF(ee)

1 + 1 -1 -1 -1 +1 + DIFF(e)

1 + 1 -1 -1 -1 +1 + 1 + DIFF($\varepsilon$)

1 + 1 -1 -1 -1 +1 + 1 + 0 = 1

deferred operations

```
DIFF(s)
  if LENGTH(s) = 0
    return 0
  if s = vs' and v is vowel
    return 1 + DIFF(s')
  else
    return -1 + DIFF(s')
```

- **Symbol $\varepsilon$ (epsilon) indicates the empty string (LENGTH($\varepsilon$) = 0)**

# Question 2: another solution

- **We use tail recursion**
  - s is a string and acc is an integer
  - Output: integer denoting number of vowels in s minus number of consonants in s

```
DIFF(s)
  return DIFF-AUX(s, 0)        // auxiliary function
```

```
DIFF-AUX(s, acc)  // input string and accumulator
  if LENGTH(s) = 0                  // empty string
    return acc
  if s = vs' and v is vowel        // v is s[0]
    return DIFF-AUX(s', acc + 1)
  else
    return DIFF-AUX(s', acc - 1)
```

# Question 2: another solution (cont.)

- **We write explicitly the recursion trace for DIFF-AUX(aabcdee, 0) (vowels in green):**

  DIFF-AUX(aabcdee, 0)

  DIFF-AUX(abcdee, 1)

  DIFF-AUX(bcdee, 2)

  DIFF-AUX(cdee, 1)

  DIFF-AUX(dee, 0)

  DIFF-AUX(ee, -1)

  DIFF-AUX(e, 0)

  DIFF-AUX($\varepsilon$, 1) = 1

```
DIFF-AUX(s, acc)
  if LENGTH(s) = 0
    return acc
  if s = vs’ and v is vowel
    return DIFF-AUX(s’, acc + 1)
  else
    return DIFF-AUX(s’, acc - 1)
```

- **No deferred operations in this case**

# Question 3

- **Prove that if $f(n) = 3n^3 + 4n + 1$ then $f(n)$ is $O(n^3)$**

# Question 3: solution

- **By definition, f(n) is O(g(n)) if there are positive constants c and $n_0$ such that**

$$f(n) \leq cg(n) \quad \text{for} \quad n \geq n_0$$

- **In our case, f(n) = 3n³ + 4n + 1 and g(n) = n³. We need to identify suitable c and $n_0$**

$3n^3 + 4n + 1 \leq cn^3$

$3n^3 - cn^3 + 4n \leq -1$

$(c-3)\, n^3 - 4n \geq 1$

$n((c-3)n^2 - 4) \geq 1$ $\quad$ $n \geq 1$

$(c-3)n^2 - 4 \geq 1$

$(c-3)n^2 \geq 5$

$n^2 \geq 5/(c-3)$ $\quad$ Take $c = 4$

$n^2 \geq 5$ $\quad$ $n \geq$

Ceiling, round-up

**Take $n_0$ = max(1, ) = 3**