# Analysis of algorithms

**Metodický koncept k efektivní podpoře klíčových odborných kompetencí s využitím cizího jazyka ATCZ62 - CLIL jako výuková strategie na vysoké škole**

Interreg EVROPSKÁ UNIE
Rakousko-Česká republika
Evropský fond pro regionální rozvoj

Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj

VŠTE

UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

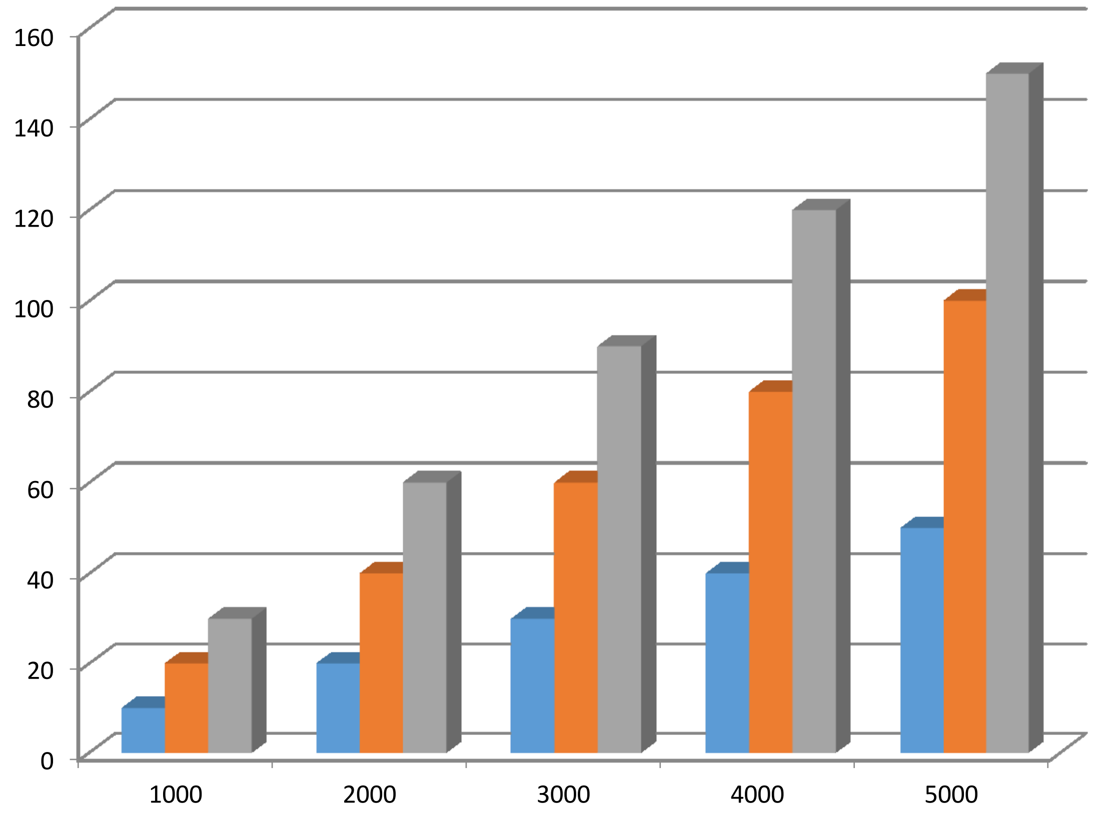
## Analysis of algorithms

- Experimental
  - Real time requirement
- Theoretical
  - Pseudo-code
  - Counting of primitive operations
  - Asymptotic notation
  - Asymptotic analysis

Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj

# Experimental analysis of time requirements

- The time requirement varies with the number of inputs and increases with the input size
- It is difficult to determine the average case
- We are focusing on the worst case scenario
  - It is easy to analyze
  - Critical for various applications
    - Games, finance, robotics, automatic operations ...,



Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj

## Experimental analysis of time requirements

- The measurement takes place in an environment where the program (algorithm) runs
- Need to implement the algorithm
  - It can be difficult
  - Requires additional knowledge
- Running depends on inputs and composition
- Not all entries are included in each run
- To compare two algorithms, it is necessary to have the same hardware and software (same running programs, same memory occupation ...)

Interreg Rakousko-Česká republika
Evropský fond pro regionální rozvoj
Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj

## Theoretical analysis

- It uses a description instead of a specific implementation
- Takes all inputs into account
- Allows you to rate the algorithm speed independently of hardware / software

## Theoretical analysis - Pseudo-code

- Higher level of algorithm description
- More structured than a classic description
- Less detailed than implementation
- Preferred Write for Algorithm Description
- It hides the problems of a specific implementation

```
Algorithm $arrayMax(A, n)$
  Input array $A$ of $n$ integers
  Output max $A$

  $currentMax \leftarrow A[0]$
  for $i \leftarrow 1$ to $n - 1$ do
    if $A[i] > currentMax$ then
      $currentMax \leftarrow A[i]$
  return $currentMax$
```

Interreg Rakousko-Česká republika Evropský fond pro regionální rozvoj
EVROPSKÁ UNIE
Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj
VŠTE
UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

# Theoretical analysis - Pseudo-code

- Running controls
  - **If … then … else**
  - **While … do**
  - **Repeat … until**
  - **For … do**
- Method (procedures, algorithm) header
  - **Algorithm** *Name (Arg1, Arg2,…)*

    **Input**

    **Output**
- Calling method (procedures, algorithm)
  - *var.Name(Arg1, Arg2,…)*
- Return values

  **return** Expression
- Expressions

| | |
|---|---|
| $\leftarrow$ | Assignment |
| = | Equality |
| +, -, $n^2$,… | Mathematical operations |

Interreg Rakousko-Česká republika
Evropský fond pro regionální rozvoj
EVROPSKÁ UNIE
Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj

## Theoretical analysis - Primitive operations

- Primitive operations
  - Basic operations performed by the algorithm
  - Identifiable in pseudo-code
  - Independent of the programming language
  - It should be precisely defined
- Příklady:
  - Evaluation of the expression
  - Assign the value to the variable
  - Indexing in the field
  - Call, return from method (procedures, algorithm)

```
Algorithm arrayMax($A$, $n$)                         Number of ops.
    $currentMax \leftarrow A[0]$                              $2$
    for $i \leftarrow 1$ to $n - 1$ do                           $2 + n$
        if $A[i] > currentMax$ then                       $2(n - 1)$
            $currentMax \leftarrow A[i]$                      $2(n - 1)$
    { increment counter $i$ }                           $2(n - 1)$
    return $currentMax$                                   $1$

                                               Total    $7n - 1$
```

Interreg Rakousko-Česká republika
Evropský fond pro regionální rozvoj
EVROPSKÁ UNIE
Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj
VŠTE
UNIVERSITY OF APPLIED SCIENCES UPPER AUSTRIA

# Theoretical analysis - Asymptotic notation

- Big O notation (Bachmann–Landau notation)
- We say that *f(n)* is *O(g(n))* for given functions *f(n)* a *g(n)*, jestliže if there is a positive constant *c* a $n_0$ such

  (   )                (   ) pro

| Notation | Name | Example |
|---|---|---|
| $O(1)$ | Constant | Determining an even / odd number |
| $O(\log n)$ | Logarithmic | Binary array sorting |
| $O(n)$ | Linear | Search in unsorted list |
| $O(n\log n)$ | Log-linear | FFT, merge sort |
| $O(n^2)$ | Quadratic | Bubble sort, borders for quicksort |
| $O(c^n)$, for $c>1$ | Exponential | Traveling Salesman Problem |
| $O(n!)$ | Factorial | Traveling Salesman Problem |

Interreg
Rakousko-Česká republika
Evropský fond pro regionální rozvoj

Europäische Union
Evropská unie
Europäischer Fonds für regionale Entwicklung
Evropský fond pro regionální rozvoj

# Theoretical analysis - Asymptotic analysis

- We specify the time-consuming algorithm using the big O notation
- We find the largest possible number of primitive operations
- Let's express them with the big O Notation
- Constants and lower order expressions can be neglected when counting primitive operations
- Example:
  - We determined that the arrayMax algorithm performs a maximum of 7n - 1 primitive operations
  - Let's say that for the arrayMax algorithm's time: $\boldsymbol{O(n)}$