Uvod v algoritmiko

Uroš Čibej

18.2.2026

Birokratski podatki

- doc. dr. Uroš Čibej, uros.cibej@fri.uni-lj.si
- predavanja 3 ure
- vaje 3 ure

Potek predmeta

- Predavanje
  - opis konceptov
  - nekaj "ad-hoc" programiranja
  - vaje na tabli
- Vaje
  - pregled nalog
  - reševanje nalog
    - programerske (na računalniku)
    - avditorne (na tablo)
- Teksti bodo na voljo na: GitHub-u

Ocenjevanje

- sprotno delo na vajah (do 20%)
- pisni izpit
- ustni izpit

Grob pregled vsebine predmeta

- Tabele (in problemi povezani z njimi)
- Kazalci
- Grafi
- Kombinatorični problemi

Literatura

1. "Data Structures and Algorithms in Python" avtorji Michael T. Goodrich, Roberto Tamassia in Michael H. Goldwasser
2. "Python Algorithms: Mastering Basic Algorithms in the Python Language" avtor Magnus Lie Hetland
3. "Grokking Algorithms: An Illustrated Guide for Programmers and Other Curious People" avtor Aditya Bhargava
4. "Data Structures and Algorithms Made Easy" avtor Narasimha Karumanchi

Dolgočasne definicije

Algoritmi + podatkovne strukture

Algoritem

- algoritem je abstraktno navodilo
- program je algoritem zapisan v nekem konkretnem jeziku
- proces je tekoči program na nekem fizičnem računalniku

Podatkovne strukture

- Način organizacije podatkov (v računalniku)
- **Abstraktna podatkovna struktura** (kaj od podatkovne strukture hočemo)
  - Primer 1. Sklad - push, pop
  - Primer 2. Vrsta - enQ, deQ
  - Primer 3. Slovar - search, insert, delete
- **Konkretna podatkovna struktura** (dejanska implementacija)
  - zaporedni podatki v pomnilniku (tabela)
  - organizacija podatkov s kazalci

Temeljna vprašanja algoritmike

- Kaj si od algoritma želim? (**specifikacija**)
- Ali algoritem dela kar si želim? (**verifikacija**)
- Kako dobro algoritem dela? (**analiza zahtevnosti**)

Specifikacija (Primer I)

```
def find(x, d)
```

- x je nek element, d je podatkovna struktura, ki hrani elemente
- Kaj si želimo od te funkcije?
  - $\forall x, \forall d : x \in d \iff find(x, d) = \text{True}$

Specifikacija (Primer II)

- ```
  def sort(a)
  ```
- Kaj si želimo od te funkcije?
  - $\forall a : b = sort(a), i < j \implies b[i] \leq b[j]$
  ```

A res?

```
def sort(a):
  return []
```

Specifikacija (Primer III)

```
def max(a):
```

Verifikacija

- preverjanje s podajanjem primerov
- preverjanje s podajanjem lastnosti
- preverjanje z dokazovanjem lastnosti

Merjenje učinkovitosti

```
def find(x,d)
```

- merjenje porabe virov
  - pomnilnik
  - *čas*
  - energija
  - komunikacija
  - ...

Merjenje časa

- čas je tipično najbolj kritičen vir
- kako bi primerjali porabo časa dveh
  - algoritmov
  - programov
  - procesov?

Eksperimentirajmo

1. izmerimo čas procesa
2. preštejmo določeno število operacij v programu
3. izračunajmo število ključnih operacij v algoritmu