Uvod u algoritme

Sadržaj

- Algoritmi u svakodnevnom životu
- Pojam algoritma
- Osnovna svojstva algoritama

Algoritmi u svakodnevnom životu

- Nakon odslušanog bit ćete u stanju:
  - navesti primjere algoritama iz svakodnevnog života
  - opisati neke jednostavne algoritme s kojima ste se već susreli tijekom školovanja
  - objasniti (ne definirati) što je algoritam
  - navesti osnovna svojstva.

Algoritmi u svakodnevnom… (2)

- Primjeri iz svakodnevnog života (upute za postizanje određenog cilja)
  - Kako doći od autobusnog kolodvora do Trga bana Jelačića:
    - čekati na istočnoj strani stajališta
    - ući u tramvaj broj 6
    - voziti se 5 tramvajskih stanica
    - izaći iz tramvaja.

Algoritmi u svakodnevnom… (3)

- Aktiviranje (nekog) aparata za gašenje požara:
  - donesi aparat na mjesto požara
  - izvuci osigurač pokretne ručice na ventilu aparata
  - dlanom udari pokretnu ručicu ventila do kraja
  - sačekaj 5 sekundi
  - uperi diznu ventila prema požaru i pritisni pokretnu ručicu do kraja
  - mlazom praha pokrivaj zapaljene površine sve dok se požar ne ugasi.

Algoritmi u svakodnevnom… (4)

- Opis postupka za
  - prelaženje ulice
  - montažu namještaja
  - povezivanje uređaja
  - puštanje u pogon perilice rublja
  - održavanje uređaja
  - …
- Postupak se sastoji od uputa. Intuitivno je jasno da upute trebaju biti
  - precizne i
  - moraju se moći izvršiti.

Algoritmi u svakodnevnom… (5)

- Primjeri algoritama s kojima ste se do sada susreli:
  - Postupak zbrajanja prirodnih brojeva:
    - napisati brojeve jedan ispod drugog tako da znamenke iste težine budu u istom stupcu
    - sa zbrajanjem se započinje s desna
    - zbrajaju se znamenke u istom stupcu i zbroju se pribraja eventualni prijenos; prijenos je na početku jednak nuli
    - ako je zbroj znamenki u stupcu ≤ 9 zapisujemo ga i prelazimo na sljedeći stupac; u protivnom zapisujemo znamenku jedinica, a prijenos postaje znamenka desetica
    - postupak se ponavlja sve dok ne zbrojimo znamenke u krajnjem lijevom stupcu; zapisujemo zbroj.

Algoritmi u svakodnevnom… (6)

- Primjeri algoritama s kojima ste se do sada susreli:
  - zbrajanje razlomaka jednakih nazivnika
  - konstrukcija trokutu upisane kružnice
  - izvođenje pokusa
  - ...
- Kada se neki postupak može nazvati algoritmom?

Pojam algoritma

Što je algoritam? Grubo rečeno:

- Algoritam = metoda, postupak, pravilo za rješenje nekog problema ili dostizanje nekog cilja.

Ovo nije precizna definicija u matematičkom smislu, već samo opis preko drugih, sličnih pojmova, pri čemu je postupak najbliži.

Pojam algoritma (2)

- Postupak asocira na konačan niz koraka koje treba napraviti za rješenje nekog problema.

- Metoda se kao izraz često koristi u matematici, ali obično uključuje i tzv. beskonačne „postupke” koji tek na limesu daju rješenje (matematička analiza, numerička matematika).

Pojam algoritma (3)

- Osnovna zadaća – razvoj efikasnih i točnih algoritama.

Intuitivno je jasno da efikasno znači brzo, a točno da je rješenje blizu „pravom” rješenju.

Osnovna svojstva algoritama

Općenito, kako moraju izgledati upute i iz čega se sastoje? Opći izgled uputa:

- 1.korak;
- 2.korak;
- …
- posljednji korak.

- Drugim riječima, algoritam se sastoji iz **konačnog** broja koraka koje treba izvršiti da bi se postigao cilj, odnosno riješio problem.

Pojam i svojstva algoritama (2)

- Postupak zbrajanja prirodnih brojeva:
  1. napisati brojeve jedan ispod drugog tako da …
  2. sa zbrajanjem se započinje s desna
  3. …
  4. …
  5. postupak se ponavlja sve dok …

upute, koraci, naredbe, instrukcije

- Svaki pojedini korak algoritma je naredba (instrukcija) koju treba napraviti (izvršiti).

Osnovna svojstva algoritama (3)

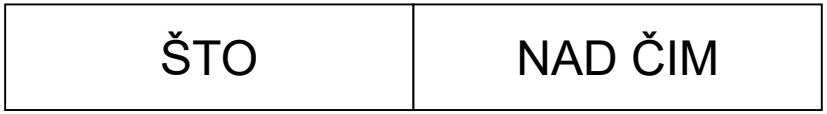
- Uobičajeno instrukcije se pišu jedna ispod druge i izvršavaju tim redom.
- Izvođenje tog niza naredbi mora biti objektivan proces koji se, u načelu, mora moći reproducirati.

- Ipak, postoje instrukcije koje mijenjaju standardni redoslijed izvršavanja: npr. „ako se dogodi to i to” onda otiđi na korak „taj i taj” ili ponovi “neke korake” sve dok je ispunjen uvjet „taj i taj”. Ovakve instrukcije su ključne u programiranju.

Osnovna svojstva algoritama (4)

Sve instrukcije imaju sličan oblik i sastoje se iz dva dijela:

- što treba napraviti i
- nad čim to treba napraviti = objekt nad kojim se obavlja operacija.
- Isti oblik imaju i instrukcije na računalu:



- Ista instrukcija se može obavljati nad raznim podacima.

Osnovna svojstva algoritama (5)

- Algoritam bi trebao raditi nad „općenitim” podacima, tj. bitno je da se samo radi o istom postupku.

- Na primjer, puno je bolje napisati algoritam koji nalazi rješenja kvadratne jednadžbe

$$ax^2+bx+c=0$$

za proizvoljne $a,b,c \in R$, $a \neq 0$ nego onaj koji nalazi rješenja jednadžbe

$$x^2-3x+2=0.$$

Osnovna svojstva algoritama (5)

- U slučaju već spomenute kvadratne jednadžbe, rješenja su dana formulom:

$$x_{1,2}=(-b\pm\mathrm{sqrt}(b^2-4ac))/(2a).$$

- Primijetite da izračunata rješenja mogu biti kompleksna.

- Što su ovdje instrukcije?

Svojstva algoritama – sažetak

Svojstva algoritma su:

- ima (ili nema) ulazne podatke
- ima izlazne podatke
- završava u konačnom vremenu
- uvijek je nedvosmisleno definiran
- mora biti efikasan (završava u razumnom vremenu).

Ulaz/izlaz

Ulaz:

- Svaki algoritam ima `0` ili više, ali konačno mnogo ulaznih podataka.
- Njih moramo izabrati iz neke dozvoljene klase ulaznih objekata.
- Algoritmi s `0` ulaza nisu česti, ali postoje. Opisuju fiksni postupak. Recimo: provjeri je li `327` prost broj ili riješi konkretnu kvadratnu jednadžbu.

Ulaz/izlaz (2)

Ulaz:

- Ako algoritam ima više različitih objekata na ulazu, kažemo da je općenit jer rješava cijelu klasu problema. Recimo: kvadratna jednadžba s općim koeficijentima `a`, `b`, `c`.

Izlaz:

- Svaki algoritam mora imati barem jedan izlaz jer inače nije ostavio trag svog izvršavanja.

Konačnost

Konačnost:

- Svaki algoritam mora završiti u konačno mnogo koraka za svaki ulaz.

- U programu uvijek treba provjeriti je li ulaz korektno zadan. Na primjer, u kvadratnoj jednadžbi koeficijent `a` može biti jednak `0`. U tom slučaju jednadžba nije kvadratna, a u formuli za rješenje pojavit će se dijeljenje s `0`!

Definiranost i nedvosmislenost

- U praksi treba predvidjeti sva moguća korisna ograničenja i ugraditi ih u program. Na primjer, program koji očitava temperaturu `T` vode u posudi trebao bi imati ograničenje da je $0 \leq T \leq 100$.

Definiranost i nedvosmislenost

- Kad projektiramo algoritam, ne znamo odmah na početku od kojih koraka se sastoji čitav postupak za rješenje problema.

Definiranost i nedvosmislenost (2)

- Algoritam se sastoji od niza osnovnih (elementarnih, primitivnih) instrukcija i mora biti jednoznačno i nedvosmisleno definiran za izvršitelja algoritma.
- Kada za rješavanje problema koristimo računalo, potrebno je znati što ono zna i može napraviti (složenost instrukcije ovisni o izvršitelju).

Efikasnost

Efikasnost:

- Algoritam mora završiti u razumnom vremenu što je bitno jači zahtjev od konačnosti – 500 godina nije razumno vrijeme!
- Na primjer, za tzv. $\mathcal{NP}$–potpune probleme ne postoje efikasni algoritmi. Postoje i problemi za koje ne postoji algoritam za njihovo rješavanje – tzv. algoritamski nerješivi problemi.

Principi rada računala

Sadržaj

- Instrukcije
- Ulaz, izlaz, memorija
- Izvršni dio računala
- Von Neumannov model
- Regularni izrazi (na vježbama)

Algoritmi u svakodnevnom životu

- Nakon odslušanog bit ćete u stanju:
  - objasniti princip rada računala
  - opisati von Neumannov model računala.

Instrukcije

Što je računalo?

- Računalo = stroj za izvršavanje algoritama
- Prisjetimo se općeg oblika instrukcije:

što = operacija nad čim = podatak ili operand

- Operacije koje računalo izvršava su osnovne strojne instrukcije.

Instrukcije (2)

- Podaci s kojima računalo „zna” raditi su osnovni ili elementarni podaci.
- Želja: imati računalo koje dozvoljava složene instrukcije nad složenim podacima.
- Zašto matematičar uopće mora nešto znati o principima rada i građi računala? Zato da bi ih mogao efikasno koristiti, pisati brze i točne algoritme (ograničenja proizlaze iz fizičke građe).

Ulaz, izlaz, memorija

- Budući da algoritam ima ulaz i izlaz, mora ih imati i računalo.
- Ulazni dio: čita podatke s nekog medija.
- Izlazni dio: mehanizam koji će na neki medij napisati podatke.

Ulaz, izlaz, memorija (2)

- Algoritam izvršava neke instrukcije nad podacima koje je učitao, tj. iz njih pravi nove podatke koje može više puta koristiti. Prema tome, računalo mora moći te podatke negdje spremiti – treba imati neku memoriju u koju može pohraniti podatke i iz koje može čitati.

Izvršni dio računala

- Potrebno je imati i izvršni dio koji izvršava instrukcije nad podacima.
- Dilema: treba li mi poseban stroj za svaki algoritam posebno ili je bolje imati stroj opće namjene?
- Poseban stroj izvršava samo jedan algoritam koji je tvrdo ugrađen u arhitekturu stroja. Nije besmisleno imati takav stroj!

Izvršni dio računala (2)

- Računalo opće namjene je složenije jer s algoritmom mora postupati slično kao s podacima. Svaki algoritam mora učitati, spremiti, izvršiti.
- Svako računalo opće namjene stalno izvršava jedan meta-algoritam (jer se stalno vrti – ne završava izvođenje dok ne isključimo stroj). To je ono što obično zovemo operativni sustav.

Izvršni dio računala (3)

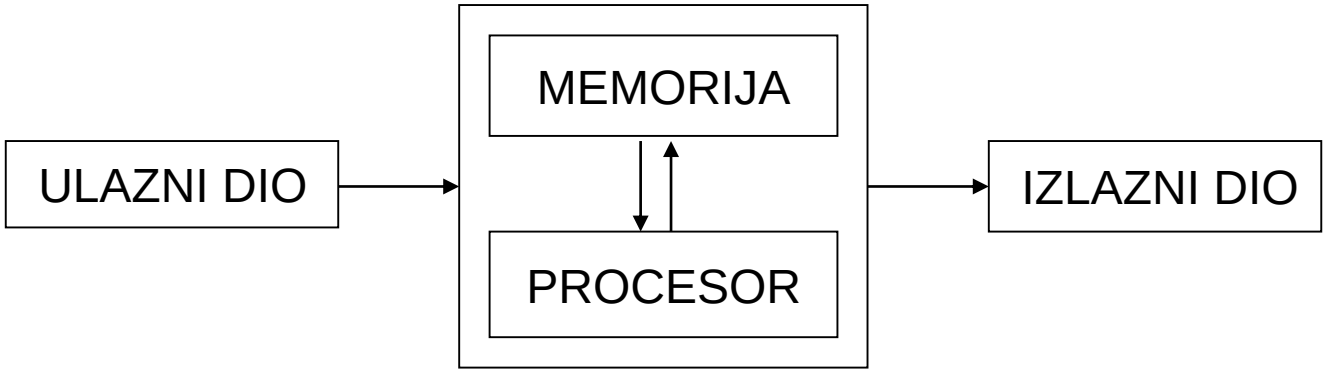
- Pitanje: gdje se spremaju algoritmi, odnosno programi?
- Mogli bismo imati posebnu memoriju za programe, ali nema potrebe za tim, pa se programi spremaju u istu memoriju kao i podaci.

Von Neumannov model

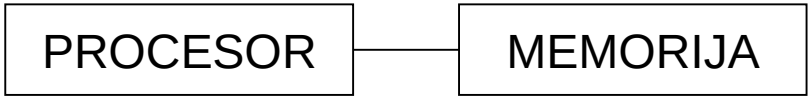
- Spremanje podataka i algoritama u istu memoriju ključna je stvar von Neumannovog modela računala.
- Ideja datira od početka 40-tih godina prošlog stoljeća, a objavljena je tek nakon 2. svjetskog rata. Model koji je on tada postavio u načelu vrijedi i za većinu današnjih računala.

Von Neumannov model (2)

- Općenito, ako izvršni dio računala zovemo standardnim imenom – procesor, onda shematski model računala izgleda ovako:



- Skraćena slika bitnog dijela računala:



ili malo detaljnije