Osnove programiranja

1

OSNOVNI POJMI
VRSTE PROGRAMSKIH JEZIKOV

Kaj je algoritem?

2

- Je nedvoumen postopek za rešitev problema v končno mnogo korakih.
- Je metoda, kako korak za korakom rešimo nalogo.

Kaj je programiranje?

- Je zapis algoritma v nekem (programskem) jeziku.

PRIMER

1. preberi vrednost x
2. preberi vrednost y
3. izračunaj vsoto x in y ter so shrani v z
4. izpiši vosto na ekran

```
1. cin >> x;
2. cin >> y;
3. z = x + y;

4. cout << "Vsota je " << z;
```


Kaj je programski jezik?

- je umetni jezik, ki je razvit z namenom:
  - da ga razume neka naprava,
  - da izraža izračune, ki jih lahko izvede naprava,
  - je skupek sintaktičnih in semantičnih pravil
- sintaksa (oz. skladnja) jezika
  - pravila, ki določajo, kako tvorimo stavke v tem jeziku
- semantika jezika
  - določa pomen posameznih stavkov in njegovih elementov

Generacije programskih jezikov

5

Programske jezike lahko delimo na 5 generacij:

1. generacija – strojni jezik
2. generacija – zbirni jezik
3. generacija – višji programski jeziki
4. generacija
5. generacija

Višja kot je generacija programskega jezika, lažje je jezik razumljiv človeku.

1. generacija – strojni jezik

6

- je najbližje računalniku
- tako zapisane ukaze računalnik neposredno razume
- programiranje je zelo zahtevno
  - velika možnost napak
  - težko odkrivanje napak
- vsak procesor ima svoj strojni jezik
  - programi niso prenosljivi med računalniki z različno strojno opremo

PRIMER:
(v namišljenem strojnem jeziku)

0001000111101010
1000101110100110
1010101011101010
1
0101010101011011
1
0101011011010101
0101010101101010
1011011011010011
1
011011001...

2. generacija – zbirni jezik

7

- izhaja iz strojnega jezika
- binarne nize, ki predstavljajo ukaze ali elemente procesorja nadomestimo z **mnemoniki**
- mnemonik je
  - kratica, ki označuje strojne ukaze
  - vsak ukaz nadomestimo z natanko enim mnemonikom
  - lažje si jih je zapomniti

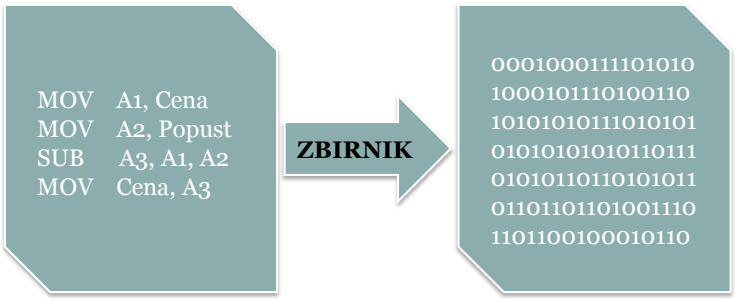
PRIMER:
(v namišljenem zbirnem jeziku)

```
MOV   A1, Cena
MOV   A2, Popust
SUB   A3, A1, A2
MOV   Cena, A3
...
```


2. generacija – zbirni jezik

8

- programa zapisanega v zbirnem jeziku procesor neposredno ne razume
- potrebna je pretvorba v strojni jezik
- to naredimo z dodatnim programom - **zbirnik**



3. generacija – višji programski jeziki (VPJ)

9

- ukazi so podobni stavkom naravnih jezikov
- dodani so matematični ukazi
- en ukaz se izvede kot zaporedje **več** stavkov strojnega jezika
- programiranje je učinkovitejše in lažje
- programi so bolj razumljivi
- lažje vzdrževanje programov
- večja prenosljivost programov na drugo strojno opremo

3. generacija – višji programski jeziki (VPJ)

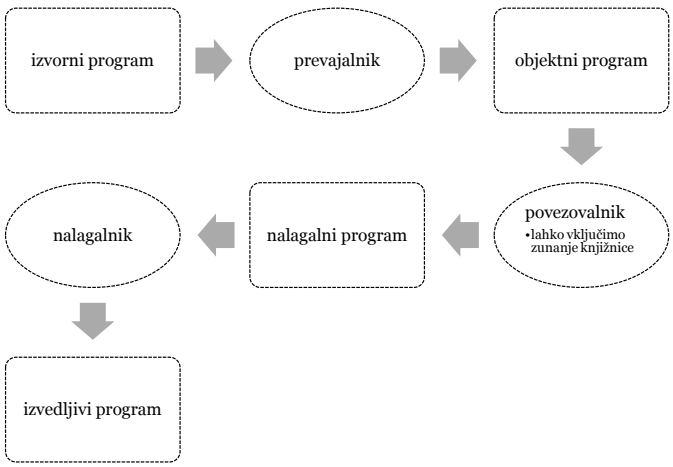
10

- program napisan v VPJ moramo pretvoriti v strojni jezik
- izvorni program >>>> izvedljivi program
- dve možnosti:
  1. prevajalnik
  2. tolmač

Prevajalnik

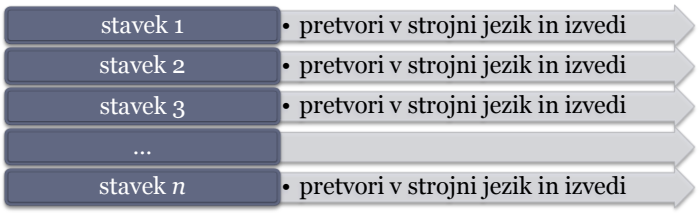
- je program, ki pretvori izvorni program (običajno zapisan v nekem VPJ) v drug (običajno strojni) jezik
- rezultat je izvedljivi program
  - računalnik ga razume
  - računalnik ga zna izvesti
- prevedemo celoten program in ga nato izvedemo

Kompleksnejša shema prevajanja



Tolmač

- je program, ki izvede izvorni program
- **stavke** izvornega programa prevede v strojni jezik in te ukaze posreduje procesorju
- izvajanje tolmačenega programa je običajno počasnejše
- prednost je interaktivnost oz. odzivnost (takoj vemo, kaj smo naredili)



3. generacija – višji programski jeziki

- večina VPJ ima prevajalnik (npr. C++)
- nekateri tolmača
- nekateri oboje (npr. Java, Python)
  - izvorni program se prevede v vmesno kodo (angl. byte code)
  - vmesna koda se tolmači s posebnim tolmačem – t.i. navidezni stroji (angl. virtual machine)
  - PREDNOST: visoka prenosljivost programa
- v VPJ zapisujemo algoritme

Programski jeziki 4. generacije

15

- omogočajo nepomembnost zapisa postopka
- cilj jezikov te generacije

  "omogočiti programerju komuniciranje s pomočjo abstraktnih pojmov na način, ki je primerljiv z načinom razmišljanja ljudi, ko rešujejo nek problem"

- PRIMER:

```
SELECT *
FROM Studenti
WHERE PovrecnaOcena>8.00
ORDER BY priimek;
```


Programski jeziki 4. generacije

16

- programer običajno ne rabi zapisati algoritma
  - lahko se osredotoči na rezultat
  - pomembno je KAJ želi rešiti in ne KAKO
- običajno niso SPLOŠNO NAMENSKI
- omejeni so na določeno področje:
  - Podatkovne baze in informacijski sistemi,
  - Generiranje poročil
  - Izdelava spletnih strani
  - Izdelava grafičnih uporabniških vmesnikov

Programski jeziki 4. generacije

17

PREDNOSTI

- so bližje programerju
- lažje programiranje
- manjša možnost napak
- enostavno vzdrževanje

SLABOSTI

- omejeni na določeno področje
- potreben je zahteven razvoj
- zahtevajo zmogljivo opremo
- so slabo učinkoviti

Programski jeziki 5. generacije

18

- mnenja se razlikujejo

1. možnost

NARAVNI JEZIK

trenutno težko uresničljiv cilj

zakaj?

2. možnost

programski jeziki 5. generacije so nekateri deklarativni jeziki

npr. Prolog

Kaj zahteva programiranje?

- programiranje je ustvarjalen proces
  - ni točnega navodila, kako programirati
- običajno poteka v dveh fazah
  - 1. faza (težji del) – reševanje problema
    - rezultat te faze je algoritem, ki reši dani problem
  - 2. faza – implementacija algoritma
    - algoritem pridobljen v prejšnji fazi zapišemo v izbranem programskem jeziku

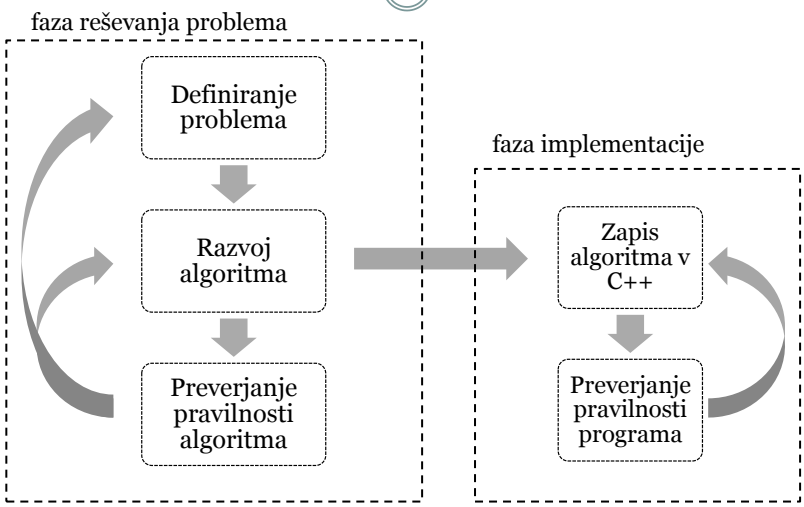
Faza reševanja problema

- natančno preberi problem
  - popolno razumevanje problema je ključno pri iskanju pravilne rešitve
- preveri, ali je problem natančno definiran
  - Kateri podatki so podani s problemom?
  - Kaj je želena rešitev problema oz. cilj?
  - V kakšni obliki moram podati rešitev?
- ne preskoči te faze – ne prični takoj z implementacijo
  - prihranimo na času
  - se izognemo nepotrebnim napakam
  - preveri, ali algoritem deluje pravilno

Faza implementacije

- algoritem dobljen v predhodni fazi pretvori v programski jezik
  - vaja dela mojstra – ta proces postaja s časom vedno lažji
- izvorni program (kodo) prevedi s prevajalnikom
  - to bo preverilo tudi, ali se držimo sintaktičnih pravil jezika
- zaženi izvedljivi program in ga testiraj z znanimi podatki
  - preveri pravilnost rezultatov
- odpravi morebitne napake

Programiranje



Integrirano razvojno okolje

- program, ki omogoča programiranje
  - podpira vse korake v procesu programiranja
  - mi bomo uporabljali Code::Blocks
- nudi vse možnosti razvoja programske opreme
  - urejevalnik
  - prevajalnik
  - povezovalnik
  - nalagalnik
  - razhroščevalnik
  - ...