# PROGRAMSKI JEZIKI

- Programski jezik je skupek:
  - Pravil
  - Podatkovnih struktur
  - Programerskih vmesnikov,
    ki programerju služijo kot orodje
    za izdelavo programa

- Računalnik se odziva le na
  2 simbola (0 ali 1 oz. DA in NE)

- Za človeka nenavadno in
  nesprejemljivo sporočanje

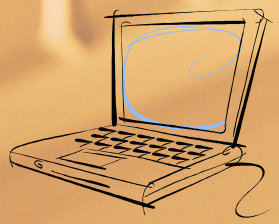
- Doslej so pripravili veliko tako im. programskih jezikov, ki se jih je mogoče razmeroma zlahka in hitro naučiti in so človeku bližji kot računalnikov strojni jezik

- Z računalnikom lahko komuniciramo le s posebno vrsto jezika, ki ga imenujemo **STROJNI JEZIK**.

- Zelo zahtevno, težavno in zapleteno opravilo za programerje in druge uporabnike

- Pisanje programov v strojnem jeziku ne pomaga širiti uporabe računalnikov
- Razloga sta bila predvsem 2:
  - Pisanje programov z zaporedjem ničel in enic je tako nepregledno, da na tak način lahko rešujemo le najpreprostejše probleme;

  - Strojni jezik se pogosto razlikuje glede na vrsto računalnikov, kar pomeni nezdružljivost (neprilagojenost, medsebojno neusklajenost, neprenosljivost) narejenih programov.



# Kako je sestavljen strojni jezik?

- Sestavlja ga niz strojnih instrukcij, vsaka izmed njih sestoji iz niza dvojiških številk (cifer) določene dolžine

- Ta niz se deli na 2 dela:
- **OPERACIJSKEGA** – predstavlja kodirano obliko določene elementarne operacije

- **NASLOVNEGA** – predstavlja kodirano obliko efektivnih naslovov v spominu in v registrih, na katerih vsebinah se želi opraviti določena elementarna operacija

- Simbolične programske jezike, ki jih delimo v 2 skupini:
- **Nižje** programske jezike (usmerjeni k stroju)

- **Višje** programske jezike (usmerjeni k problemu)

- Predstavnik nižjih programskih jezikov – t.i. **ZBIRNI JEZIK**, ki strojne instrukcije predstavlja v simbolični ali številčni obliki in se ne more neposredno izvajati na računalniku, saj ga je treba prej prevesti v strojni jezik

```
format PE GUI 4.0
include 'win32ax.inc'
.code
        invoke
MessageBox,HWND_DESKTOP,"Zivijo
svet!«,"Program Zivijo Svet«,MB_OK
        invoke  ExitProcess,0
```

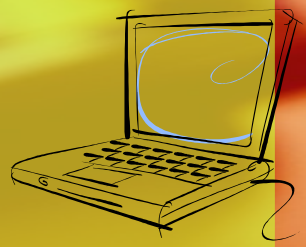
Primer »živijo svet« programa v Ploščatem Zbirniku

# Kako poteka pisanje programov v teh jezikih?

- Program napišemo v zbirnem jeziku
- Uporabi se kot vhodni ali izvirni program v **ZBIRNIKU** (program, ki prevede program v zbirnem jeziku v strojni jezik in preveri, ali so uporabljeni pravilni nazivi in oblike ukazov)
- Če program nima napak, zbirnik oblikuje izhodno datotek (ciljna koda)
- To je sedaj program, preveden v str nko izvede na računalniku

- Pisanje programov v zbirnem jeziku enostavnejše kot pisanje s strojnim jezikom
- Pomanjkljivosti in prednosti
- **PREDNOSTI**:
- Izkušenemu programerju nudi veliko možnosti za natančen nadzor nad uporabo, za ukaze in za doseganje hitre izvedbe
- Uvedba **MAKROJEV** – okrajšave za standardne nize osnovnih ukazov; Programerjem olajšujejo delo v zbirniku tam, kjer bi se moral niz ukazov ponoviti na več mestih programa

- Vključi se makroukaz, ki rešuje programerje, kadar ponavljajo pisanje enih in istih skupin ukazov, in zmanjšuje možnost, da bi delali napake



- Okoli leta 1950 so veliko razpravljali o uporabi ustreznih programskih jezikov;
- Da bi se programiranje poenostavilo in približalo človeku, so se pojavile nove vrste jezikov – **VIŠJI PROGRAMSKI JEZIKI**

- Bližje človekovemu načinu zapisovanja
- Problemsko usmerjeni – njihova struktura prirejena področju, za katerega so namenjeni
- Potrebujejo prevajalnike, ki skrbijo za »razumevanje« med zapisanim programom v višjem programskem jeziku in strojnim ukazom, ki ga razume računalnik

- Za pretvorbo višjega prog. jezika strojni jezik se uporabljata 2 načina:

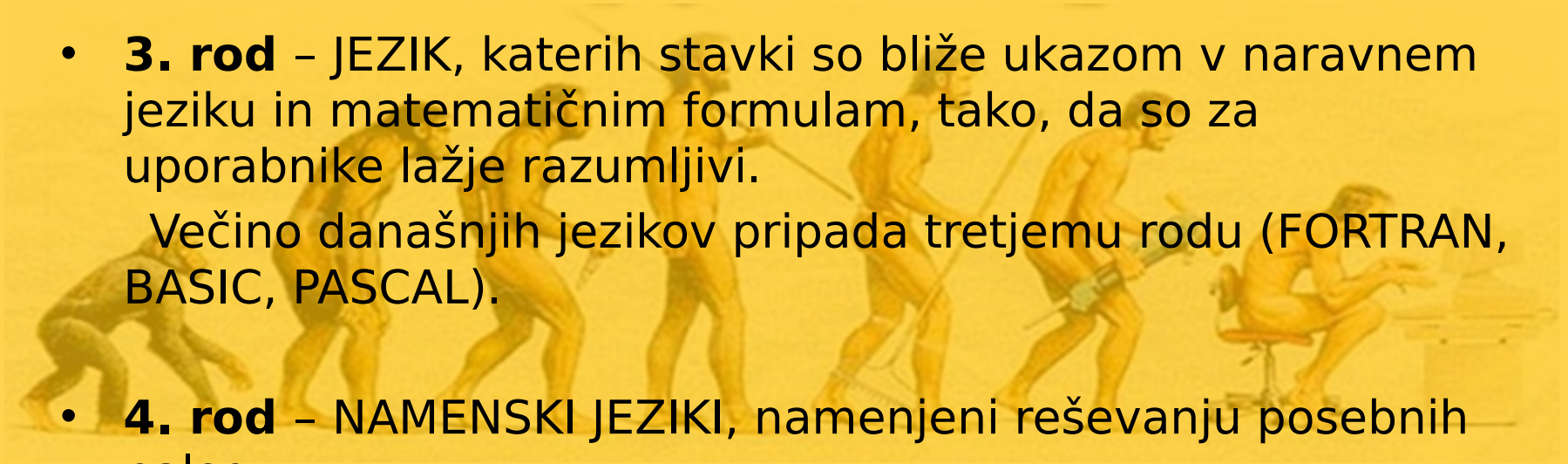
- **PREVAJANJE** – poseben program, ki celoten program prevede v strojni jezik računalnika

- **TOLMAČENJE** – poseben program, ki pri izvajanju programa sproti pregleduje, urejuje, javlja napake in prevaja oz. tolmači posamezne stavke programa.
- Ponavadi okoli 10x počasnejše kot izvajanje prevedenega programa

# ZGODOVINSKA DELITEV

- **1. rod** – STROJNI JEZIK
- **2. rod** – ZBIRNI JEZIK
- **3. rod** – JEZIK, katerih stavki so bliže ukazom v naravnem jeziku in matematičnim formulam, tako, da so za uporabnike lažje razumljivi.

  Večino današnjih jezikov pripada tretjemu rodu (FORTRAN, BASIC, PASCAL).

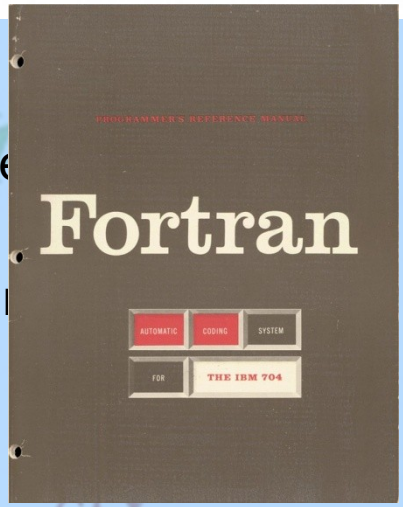


- **4. rod** – NAMENSKI JEZIKI, namenjeni reševanju posebnih nalog:
  - vzdrževanje zbirk podatkov
  - matematične operacije
  - ali opisi zapletenih listin

- **5. rod** – JEZIKI INTERNETA, INFORMACIJSKE DRUŽBE in

# POMEMBNEJŠI PREDSTAVNIKI

- **FORTRAN** (1955)
  - Nastal kot program za prevajanje mate
    postopkov v računalniške programe
  - Uporablja se predvsem za reševanje zn
    tehničnih problemov
  - Več različic




- **COBOL** (1960)
  - Nastal 1960, vpeljan pa 7 let kasneje
  - Namenjen pisanju programov za poslovno-ekono
    problematiko
  - Narejen tako, da skuša biti čim bolj podoben ang
    in razumljiv tudi neprogramerjem, npr. vodstve

- **BASIC** (1965)
  - Nastal kot preprost jezik za učenje programiranja in kot stopnička k fortranu
  - Zasnovali so ga kot jezik za pogovorno, interaktivno delo z računalnikom in kot jezikom, ki bo primeren za najrazličnejše uporabnike: začetnike, študente, tudi strokovnjake

Microsoft Visual Basic for Applications

- **PASCAL** (1968)
  - 1. prevajalnik je nastal 2 leti kasneje
  - Imenovan po francoskem matematiku Blaisu Pascalu, ki je izumil enega prvih praktično uporabnih računskih strojev
  - Pascal se je hitro razširil po svetu in v nekaj letih postal eden pomembnejših programskih jezikov

Pascal

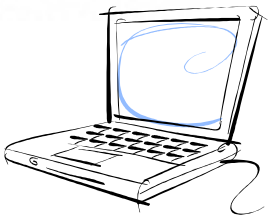
- **C** (1983)
  - Zelo sposoben višji programski jezik
  - Posebno primeren za pisanje operacijskih sistemov in sistemskih servisnih programov

C Programming

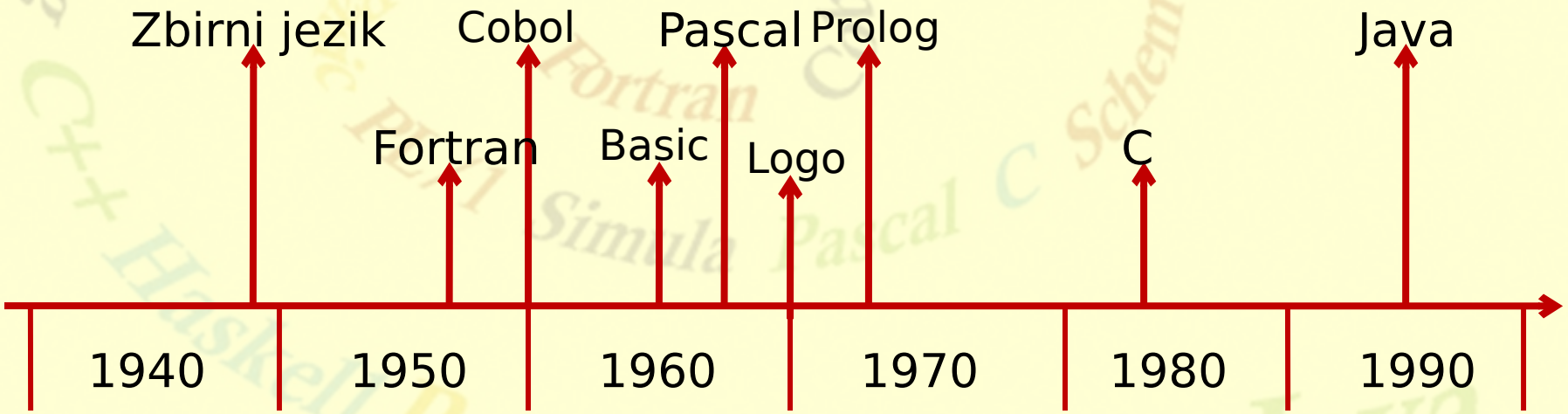
- **LOGO** (1970)
  - Izpeljanka iz grške besede LOGOS (gr. logos – misel, razum)
  - Njegovi avtorji so želeli ustvariti jezik, ki bi povezoval način programiranja in način človekovega razmišljanja
  - Razvijal več let, s sodelovanjem strokovnjakov z obeh strani Atlantika
  - Uporaba po vseh šolah, zlasti v nižjih razredih osnovne šole

- **JAVA** (1995)
  - Najnovejši med vsemu prestavljenimi
  - Primeren za pisanje programa za na internet in druga omrežja
  - Primere takih programov najdemo v obliki:
    - Kratkih animacij
    - Mini iger
    - Slik z atraktivnim odzivom

- **PROLOG** (1973)
  - Jezik za logično programiranje, ki ga uporabljajo s področja *umetne inteligence* (veja računalništva, ki se ukvarja z razvojem programov)



# ČASOVNICA RAZVOJA PROGRAMSKIH JEZIKOV



- Več kot 100 višjih programskih jezikov

- Delimo jih v 2 vrsti:
  - jezike, s katerimi predvsem opisujemo, kaj želimo narediti
  - jezike, s katerimi tudi podrobneje povemo, kako to storiti

- Jeziki 1. vrste – **NEPOSTOPKOVNI** ali **OPISNO USMERJENI**

  - Po svoji strukturi in logiki so najbolj prilagojeni človeškemu načinu razmišljanja
  - Pomemben predstavnik je PROLOG

- Jezike 2. vrste im. **POSTOPKOVNE** ali **UKAZOVALNO USMERJENE**

  Opisujejo program podobno kot kuharska knjiga kuho: vzemi to ali to, stori prvo, drugo, tretje …

# VNOS, TESTIRANJE IN OPREMA PROGRAMA

- PISANJE PROGRAMOV je večstopenjski proces. Razdelimo na 5 korakov:
  - Analiza programa
  - Zasnova programa
  - Pisanje programske kode
  - Preskušanje programa in odpravljanje napak
  - Dokumentiranje programa

- Računalnik sam po sebi ne naredi ničesar:
  - Če mu zapovemo, naj naredi neumnost, jo bo zagotovo zagrešil
  - Če mu narekujemo napačen način reševanja problema, ga bo po napačni poti tudi izpeljal
  - Če želimo torej z računalnikom rešiti kak problem, moramo postopke reševanja vnesti natančno in strokovno pravilno, s pomočjo algoritma

- **ALGORITEM** – spisek navodil, ki nas v končnem številu korakov privedejo do rezultata

- 1. Ko z algoritmom rešimo določen problem, je treba rešitev kot delovno nalogo prenesti v računalnik v izbranem programskem jeziku

- 2. Poskusno ga pregledamo in hkrati popravimo glavne napake
- Obravnava napak v programu je za računalnik zapleten in zahteven postopek,
- v katerem ni mogoče predvideti vseh napak, ki jih zagreši programer

- 3. Treba je poskrbeti, da je program napisan pravopisno pravilno

- Programske napake delimo v:
  - Pravopisne
  - Logične
  - Snovalske

- **PRAVOPISNE** – nastanejo zaradi slovnično nepravilnih zapisov ukazov
- **LOGIČNE** – zahtevnejše, saj zaradi njih program ne deluje pravilno
- **SNOVALSKE** – najhujše, zahtevajo veliko premišljevanja in popravljanja (če npr. v zasnovi uporabimo nepravilen algoritem ali algoritem z napakami)

- Za popoln opis jezika je potrebno:
- **Opis zgradbe jezika** – vedeti moramo, kako so zgrajeni pravilni programi
- **Opis pomena**, ki pove, kaj naj sintaktično pravilen program počne

- 4. Ko so osnovne programske in pravopisne napake odpravljene program shranimo na ustrezen element zunanjega spomina (disketo, zgoščenko)

- 5. Dobimo 1. verzijo programa, uporabnost programa je potrebno še preveriti

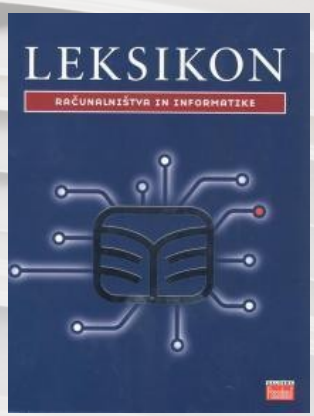
- 6. Postopek testiranja navadno poteka tako, da v program vnašamo izmišljene podatke, težavnost, zapletenost in različnost podatkov pa stopnjujemo

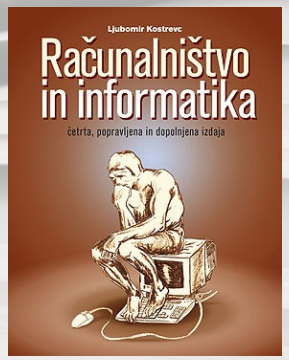
- Na ta način ugotavljamo, ali so vrednosti, ki jih dobimo kot rezultat procesa, pravilne in logične

- Na koncu je treba programsko dokumentacijo dodelati, opraviti poskusno dobo in poskrbeti za trajno hrambo originala programa
- Program, ki je testiran, moramo še nekaj časa preizkušati
- Program nato shranimo na element zunanjega spomina (disk, disketa, zgoščenka) v vsaj 2 verzijah
- Dober in skrben programer zbrano dokumentacijo na koncu uredi, nato pa napiše še podrobna navodila za delo in uporabo programa, saj s tem zelo pomaga bodočim uporabnikom

# VIRI

- Knjižni:
- Leksikon računalništva in informatike
  - Avtorji: David Pahor, Matija Drobnič, Vladimir Batagelj
- Računalnik za vsak dom
  - Avtorja: Mateja in Sašo Zorman
- Računalništvo in informatika
  - Ljubomir Kostrevc




- Internetni:
- http://splet-stari.fnm.uni mb.si/pedagoska/didgradiva/2005/eucbenik2/poglavje4_3.htm
- http://sl.wikipedia.org/wiki/Seznam_programskih_jezikov
- Google slike