Procesorski sistemi v telekomunikacijah
Programiranje, prevajanje in izvajanje programov

Strojni in zbirni jezik

Naslov	Vsebina (HEX)
...	
a	A3
a+1	00
a+2	A0
a+3	E3
a+4	12
a+5	00
a+6	90
a+7	E2
...	

```
MOV R0, #0xA3
ADDS R0, R0, #0x12
```

strojni jezik

zbirni jezik

- Strojni jezik: številske kode ukazov in njihovih operandov
- Zbirni jezik (assembly language): besedni zapis za strojni jezik
- Preslikava enega v drugega je preprosta.
- Program za preslikavo zbirni jezik v strojni jezik: **zbirnik (assembler)**
- Program za preslikavo strojnega v zbirni jezik: **disassembler**

- Disassembler se pogosto uporablja za tki. reverzni inženiring - študiranje tujih programov, katerih izvorne kode (zbirne kode) nimamo na razpolago.
- Strojni jezik ARM7 pozna dva nabora ukazov: ARM in THUMB.
  Ukazi ARM načina so dolgi 32 bitov, ukazi THUMB načina pa 16 bitov.
  THUMB ukazi so okrnjeni ARM ukazi. THUMB programi so krajši.
  Preklop v THUMB način delovanja: bit T (5. bit) registra CPSR postavimo na 1.

Višji programski jeziki

- Enostavnejše izražanje koceptov, kot je naprimer zanka:

```
        MOV     R0, #0
        LDR     R1, =__bss_start__
        LDR     R2, =__bss_end__
Loop:   CMP     R1, R2
        STRLO   R0, [R1], #4
        BLO     Loop
```

```
unsigned int *i;
i=(unsigned int *)__bss_start__;
while (i<(unsigned int*)__bss_end__)
{
  *i=0;
  i++;
}
```

- Bolj zapleteni podatkovni tipi kot v strojnem jeziku.
  Npr. strojni jezik ARM7 pozna le en tip ... 32-bitna cela števila.
  Delo z realnimi števili ni enostavno, če programiramo v strojnem jeziku.
- Enostavnejše pisanje matematičnih formul.
  Poskusite napisati tole (izračun e - osnove naravnega logaritma) v strojnem jeziku ARM7 ☺

```
int i, n=20;
double clen=1.0, e=1.0;
for(i=1;i<=n;i++) {
  e=e+clen;
  clen=clen/(i+1);
}
```

Strojno razumevanje višjih jezikov
1. Besedna analiza (Lexical Analysis)

- Razpoznavanje elementov, kot so rezervirane besede, operatorji, konstante, identifikatorji, imena spremenljivk, posebni znaki) ...

```
a = 1 ;
i = 0 ;
while ( a < 10 ) {
    i = i + 1 ;
    a = a * i ;
}
```

posebni znaki

operator

konstanta

rezervirana beseda

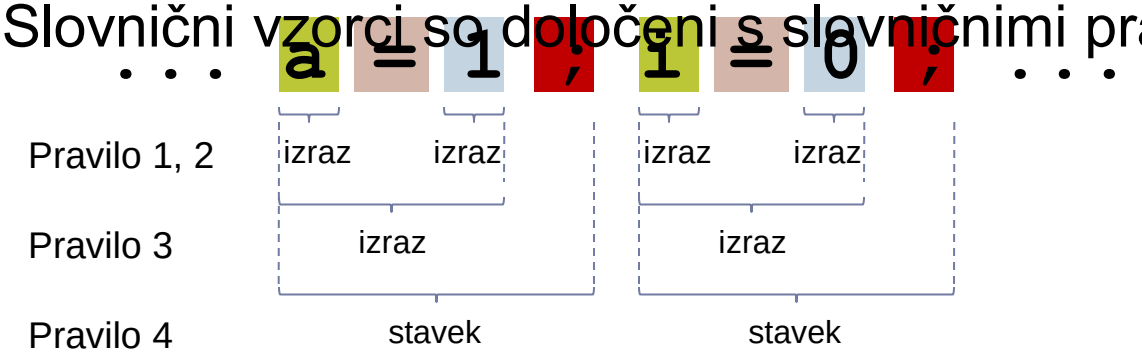
identifikator

- Razpoznavanje je enostavno, saj se programski jeziki strogo držijo predpisanih pravil, teh pa je malo in so preprosta.
- Rezultat je zaporedje osnovnih simbolov (angl. token).

```
...; i = 0 ; while ( a < 10 ) { ...
```

Strojno razumevanje višjih jezikov
2. Slovnična analiza (Parsing)

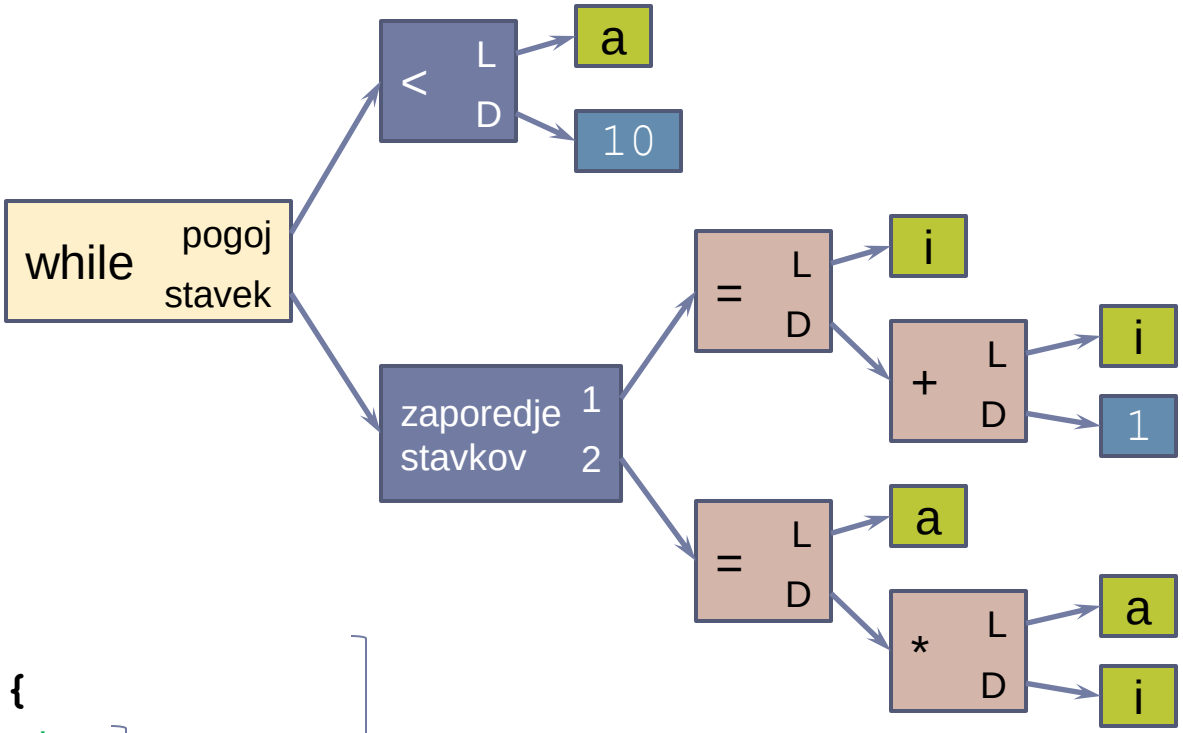
- Razpoznavanje slovničnih vzorcev v zaporedju osnovnih simbolov
- Slovnični vzorci so določeni s slovničnimi pravili



- Pravilo 1: **konstanta** == **izraz**
  identifikator == **izraz**
  Pravilo 2: **izraz operator izraz** == **izraz**
  Pravilo 3: **izraz podpičje** == **stavek**
  Pravilo 4: **stavek stavek stavek ...** == **zaporedje stavkov**
- Razpoznavanje vzorcev v zaporedju simbolov je enostavna naloga, ker so pravila ponavadi enostavna in jih ni veliko.

Rezultat slovnične analize: Drevo izpeljav (Syntax Tree)

- Razpoznane slovnične vzorce se da predstaviti v obliki drevesa.




```
while ( a < 10 ) {
  i = i + 1 ;
  a = a * i ;
}
```

pogoj (izraz)

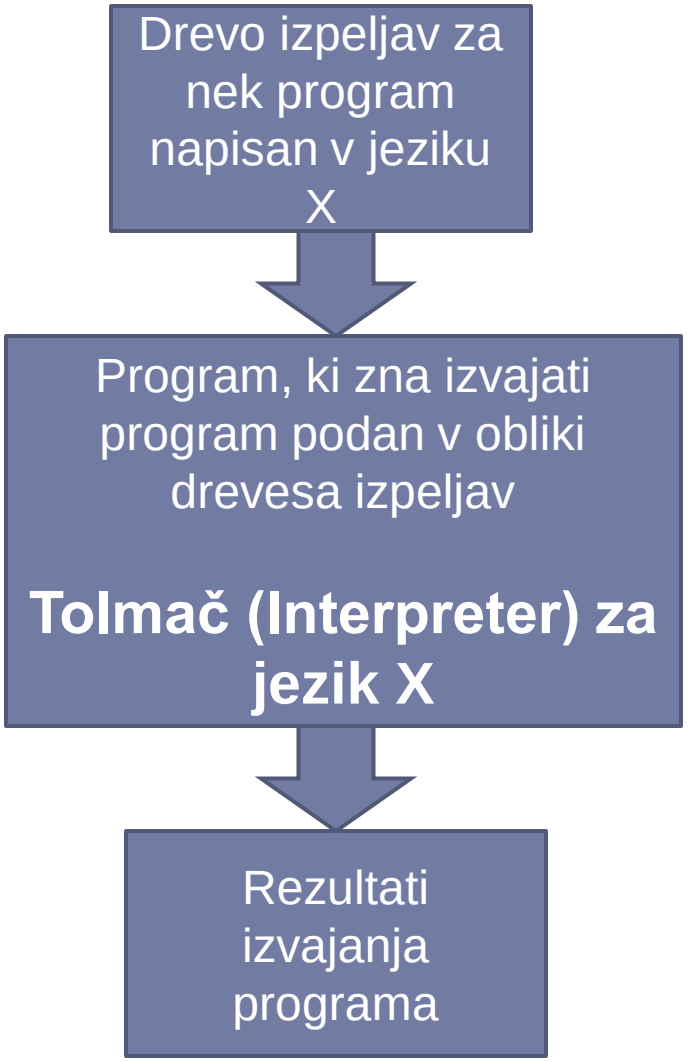
stavek

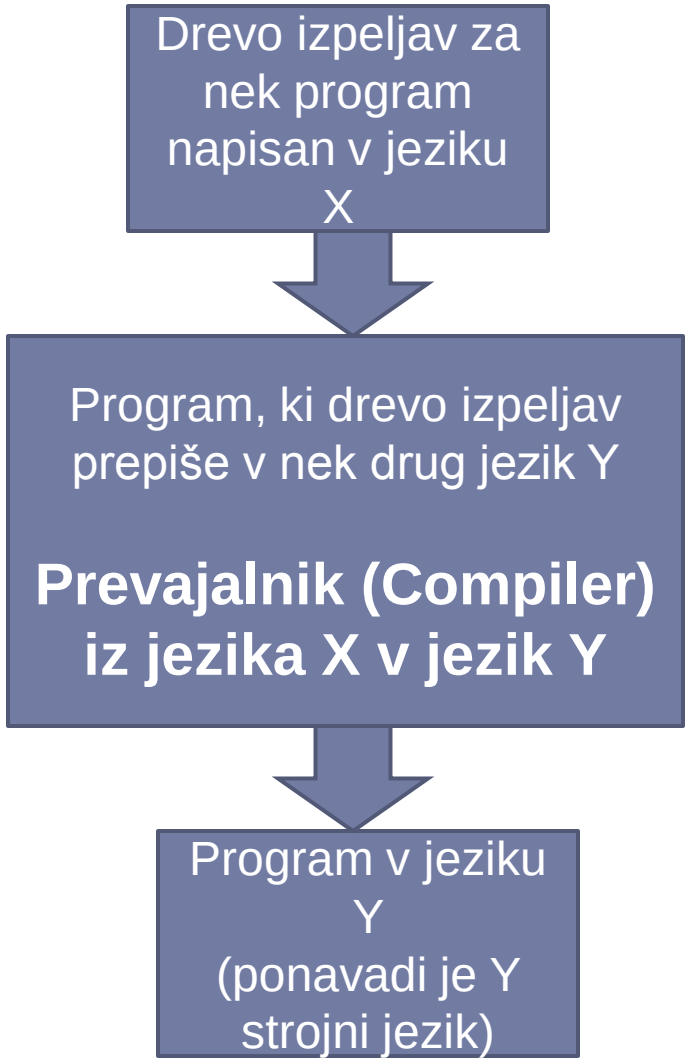
stavek

zaporedje stavkov

zanka WHILE

Kaj početi z drevesom izpeljav?





Tolmač (Interpreter)

- Je program za izvajanje programov v nekem jeziku A.
- Pred izvajanjem se program iz jezika A prevede v drevo izpeljav.
- Včasih temu sledi še prevajanje v bolj kompaktno obliko – byte-kodo (Bytecode), ki se nato izvaja v posebnem tolmaču –

  navideznem stroju (Virtual Machine)
- Nekateri tolmači izvajajo program brez, da bi prej zgradili drevo izpeljav oz. byte-kodo (npr. nekatere preproste ukazne lupine).
- Byte-koda se lahko shrani v posebno datoteko. Ob naslednjem zagonu programa jo navidezni stroj prebere iz datoteke.

  Primer: Python – programi so v datotekah .py, byte-koda pa v datotekah .pyc
- Program v tolmaču teče počasneje, kot bi tekel, če bi ga pred zagonom prevedli v strojni jezik

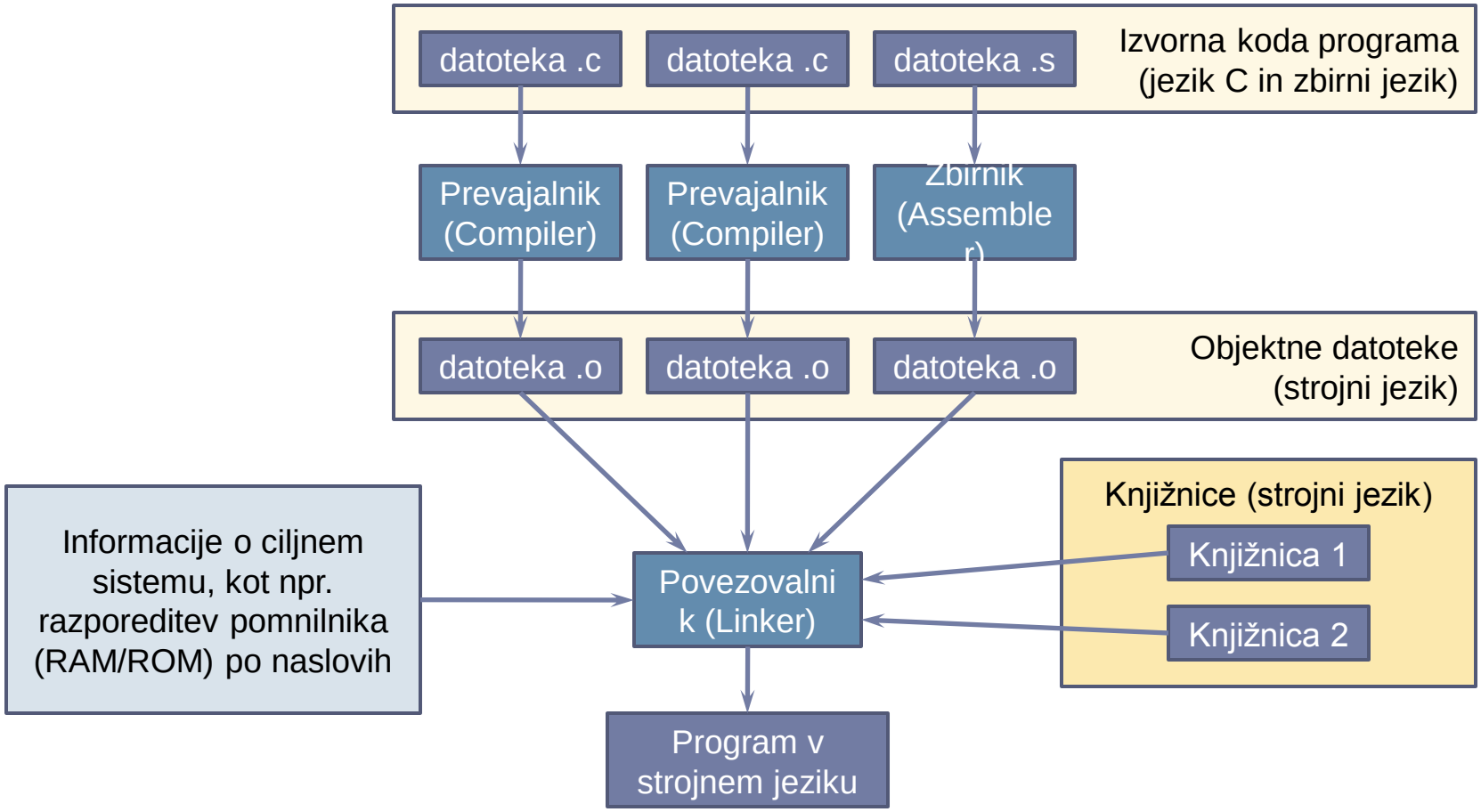
Predčasni prevajalnik
JIT (Just-In-Time) Compiler

- Tik pred izvajanjem posameznih delov programa se byte-koda prevede v strojni jezik.
- Hitrejše izvajanje, kot bi ga dobili z navideznim strojem.
- Mogoče so optimizacije programa na osnovi podatkov zbranih med delovanjem programa.
- Primer: Java, JavaScript (Firefox), .NET okolje, MATLAB, ...
- Ideja se pojavi že leta 1960 z jezikom LISP (McCarthy),
  jezik LC leta 1970 (Mitchell 1970).
  Jezik Smalltalk – različica Self (podjetje SUN)

dosegala

Statično prevajanje v strojni jezik

- Program (izvorna koda) se v celoti prevede v strojni jezik še pred izvajanjem.
- Primer: program delno napisan v jeziku C, delno pa v zbirnem jeziku



Optimizacija kode

- Prevajalnik lahko pri prevajanju obdela program tako, da je končni rezultat krajši (optimizacija velikosti) ali pa, da se izvaja hitreje (optimizacija hitrosti).
- Primer – optimizacija hitrosti:

Neoptimiziran program

```
double a, b, c, d;
/* tukaj beri a */
b=sin(cos(a)+pi/4);
c=cos(a);
d=cos(a)+pi/4;
```

Optimiziran program
(ta se dejansko prevede v strojni jezik)

```
double a, b, c, d;
/* tukaj beri a */
c=cos(a);
d=c+pi/4;
b=sin(d);
```

- Po optimizaciji se lahko spremeni vrstni red izvajanja stavkov, način hranjenja spremenljivk (registri/pomnilnik), izginejo lahko nekatere spremenljivke in stavki, ...

Statične in dinamične knjižnice

- Statične knjižnice – statično povezani programi
  dinamične knjižnice – dinamično povezani programi
- Statično povezani programi vsebujejo svojo kopijo knjižnic.
- **Dinamično povezani programi** knjižnice naložijo ob zagonu.
- Prihranek prostora na disku: programi so krajši,
  vsebina knjižnice se pojavi le enkrat (v datoteki, ki hrani knjižnico).
- Prihranek pomnilnika: če teče več programov, ki uporabljajo isto knjižnico, imamo v pomnilniku le eno kopijo knjižnice.
- Nadgradnje funkcionalnosti programov z zamenjavo knjižnice – pri statično povezanih programih je potrebno ponovno prevajanje, pri dinamično povezanih pa le menjava datoteke s knjižnico.
- Dinamične knjižnice omogočajo izvedbo vtičnikov (plug-in).
  Nalaganje po potrebi.
- Windows: .lib (statične) in .dll (dinamične) knjižnice
  Linux: .a (statične) in .so (dinamične) knjižnice

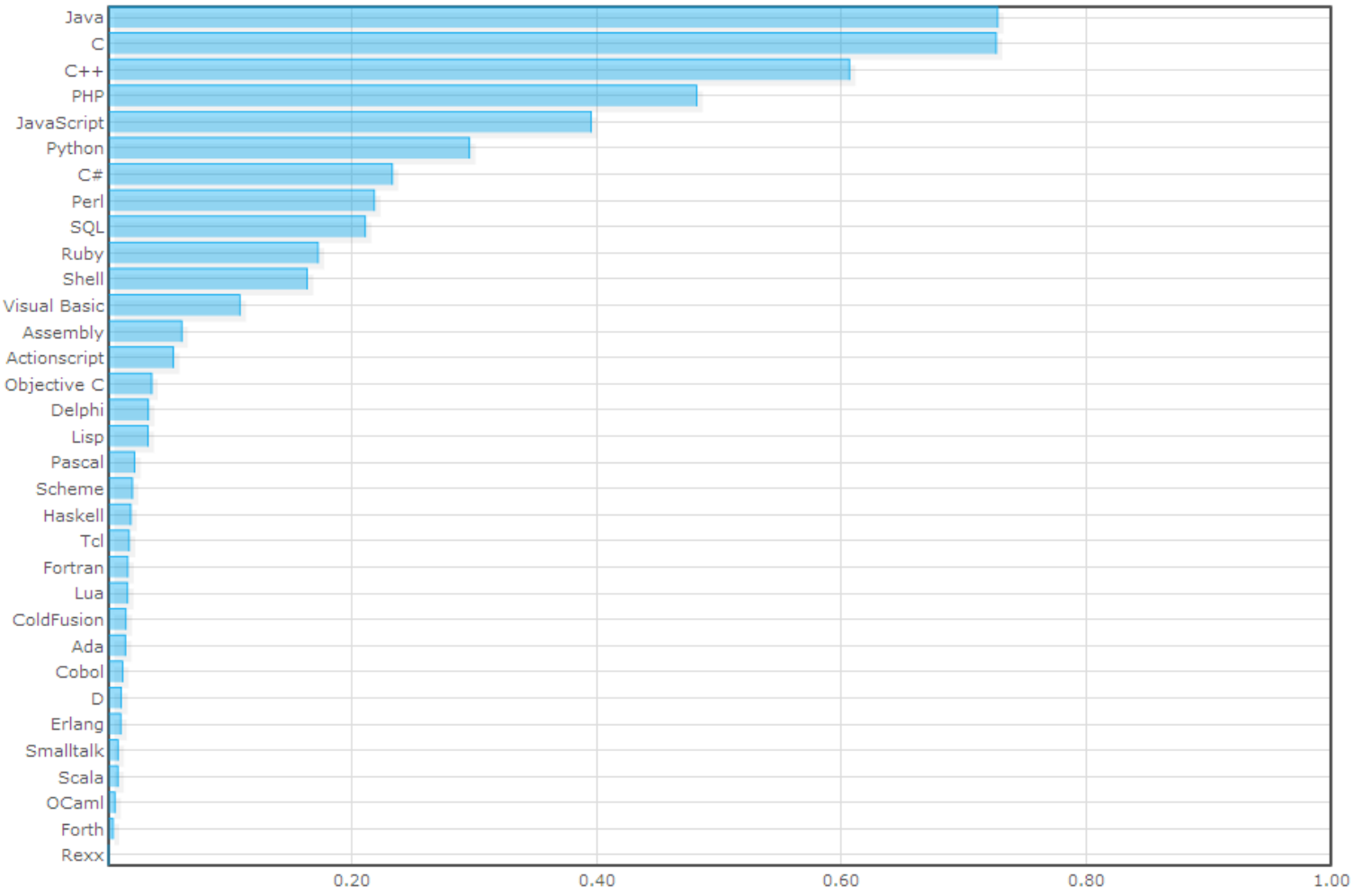
Navzkrižni prevajalnik (Cross compiler)

- Prevajanje programa ponavadi poteka na računalniku, ki bo program tudi izvajal.
- Za prevajanje programov, ki se bodo izvajali na manjših mikroprocesorskih sistemih, prevajanje običajno poteka na osebnem računalniku. Tako prevajanje opravlja **navzkrižni prevajalnik**.
- Primer: WinIdea za š-ARM teče na osebnem računalniku (i386). Za prevajanje uporablja prevajalnik **gcc**, ki teče na procesorjih družine i386. **gcc** prevaja C v strojni jezik jedra ARM7. Dobljenega programa ne moremo zagnati na osebnem računalniku, ampak le na ARM7 procesorju.

Projekt GNU

- Začetek 1983, Richard Stallman, MIT
- Cilj – brezplačen operacijski sistem in programska oprema
- Do 1992 razvito veliko programske opreme, ki tvori osnovo sistemov UNIX, med drugim tudi prevajalnik za jezik C/C++ (gcc).
- Mejnik: 1992 – jedro sistema Linux objavljeno pod licenco GPL
- Licenca GPL: določa pogoje uporabe in distribucije programov. Zelo svobodna glede uporabe, stroga glede izvorne kode: izvorna koda in vse njene spremembe morajo biti javne.
- Vsakdo, ki se ne strinja s smerjo razvoja GPL programa, lahko odcepi svojo inačico in jo razvija naprej po svoje. Licenca se ne sme spremeniti.
- Licenca LGPL - za knjižnice.
  Dovoljuje povezovanje LGPL knjižnic v programe, katerih licenca ni GPL/LGPL. Ne zahteva javnost sprememb izvorne kode, zahteva pa, da je prevedena oblika sprememb (strojni jezik) javna.

Kateri jeziki so aktualni danes ...



Vir: http://langpop.com/ (23.10.2010)

Jeziki…

- C ... od leta (Bell Labs, Dennis Ritchie,1972), skoraj zmeraj se prevaja
- C++ ... objektno orientirana (OO) razširitev C-ja (Bell Labs, Bjarne Stroustrup, 1983), pogost v velikih projektih, skoraj zmerja se prevaja
- Objective C ... še ena OO razširitev C-ja (Stepstone, Brad Cox, Tom Love, 1986), popularizira ga firma Next (Steve Jobs), danes Apple
- Java ... OO, podoben C++ (Sun, James Gosling, 1995), izvaja se s pomočjo tolmača in JIT prevajalnika (Java Virtual Machine, JVM)
- JavaScript ... OO, podoben C-ju in Javi (Netscape, Brendan Eich, 1995), tolmačen. Uporaba – dinamične spletne strani, za izvajanje programov v spletnih brskalnikih (na strani odjemalca).
- PHP ... OO, podoben C-ju (Rasmus Lerdorf, 1995), tolmačen. Uporaba – dinamične spletne strani, za izvajanje programov na strani spletnega strežnika (npr. v Apache).
- Python ... OO, (Guido van Rossum, 1991), tolmačen, modularen, veliko knjižnic.
- Perl ... OO, (Larry Wall, 1987), tolmačen, modularen, veliko knjižnic.
- C# (C sharp) ... OO, podoben C-ju in Javi (Microsoft, 2001), tolmač z JIT prevajalnikom, izvaja se v okolju CLR (Common Language Runtime),

… jeziki

- SQL .. jezik za poizvedbe v bazah podatkov (IBM, Chamberlin, Boyce, 1974)
- Ruby ... OO (Yukihiro Matsumoto, 1995), popularen postal z uporabo v okolju
  Ruby on Rails za hiter razvoj spletnih aplikacij
- Shell ... skupno ime za skriptne jezike različnih ukaznih lupin
- Visual Basic ... inačica BASIC-a (Microsoft, 1998) za hiter razvoj okenskih programov
- ActionScript ... izvedenka standarda ECMAscript (tako kot JavaScript), Macromedia, Gary Grossman, 1998), jezik za programiranje komponent Flash (spletne strani)
- Pascal ... v prvi vrsti učni jezik (Niklaus Wirth, 1970), zelo strog
- Delphi ... okolje za hiter razvoj okenskih aplikacij objektnem Pascalu (Borland, 1995)
- LISP ... eden prvih jezikov, zelo zmogljiv, težko berljiv (John McCarthy, 1958),
  današnje inačice: Scheme, Common Lisp, ...
  grajen v AutoCAD (program za tehnično risanje)
- Fortran ... eden prvih jezikov (John Backus, IBM, 1957), še danes v uporabi