Programski jezici

Programski jezik

- Programski jezik računala je jezik koji računalo razumije.
- Postoji mnogo programskih jezika, a svaki od njih ima područje primjene, prednosti i nedostatke.

Programski jezici - podjela

- Programski jezici se mogu podijeliti na:
  - strojne jezike,
  - simboličke jezike niske razine,
  - simboličke jezike visoke razine.

Strojni jezik

- Strojni jezik (engl. *machine language, machine code*) je binarni prikaz programa za računalo.
- To je ujedno i jedini oblik programa koji računalo doslovno "razumije“.

Strojni jezik

- Strojni je jezik vezan uz građu računala, odnosno ovisan je o središnjoj jedinici za obradu.
- Svaki procesor ima sebi svojstven strojni jezik.
- Pisanje programa strojnim jezikom je složeno i zahtijeva dobro poznavanje građe računala te se njime bave usko specijalizirani stručnjaci.

Simbolički jezici

- Simbolički su jezici nastali kako bi ljudima olakšali programiranje jer ljudi lakše pamte simbole nego binarne brojeve.
- Programi pisani simboličkim jezikom su čovjeku čitljiviji i lakši za razumijevanje od binarnog zapisa.

Simbolički jezik niske razine

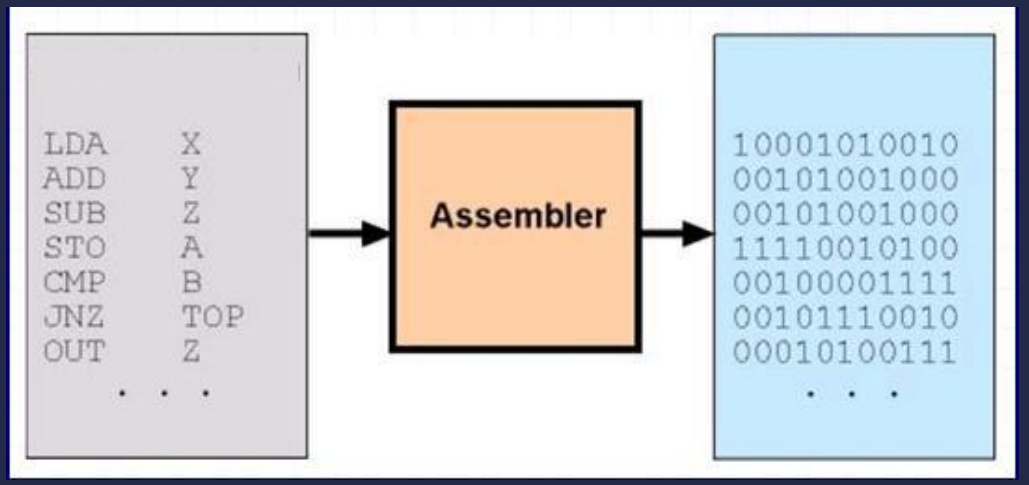
- Asembler (engl. *assembler*) je sibolički jezik niske razine u kome je svaka binarna naredba strojnog jezika predočena odgovarajućim simbolom npr.:
  - ADD
  - SUB
  - CMP

Simbolički jezik niske razine

- Svaka simbolička naredba obično je kratica engleskog opisa naredbe pa se lako pamti.
- Asembler je čovjeku čitljiviji i lakši za razumijevanje od binarnog zapisa ali još uvijek vrlo težak za pisanje i ispravljanje.

Simbolički jezik niske razine

- Program napisan u asembleru mora biti preveden u binarni oblik da bi ga procesor “razumio”, tj. da bi ga mogao izvršavati.
- Simbole u binarni oblik prevodi program prevoditelj.



Simbolički jezici visoke razine

- Simbolički jezici visoke razine su svi oni programski jezici kod kojih se veći ili manji skup naredaba strojnog jezika nadomješta jednom naredbom višeg programskog jezika.
- Naredbe ovih jezika mnogo su više nalik govornom jeziku, lakše su za pamćenje i upotrebu.

Simbolički jezici visoke razine

- Simbolički jezici visoke razine (viši programski jezici) stvoreni su da bi se:
  - olakšalo programiranje,
  - isti program mogao izvršavati na različitim računalima (procesorima).

Simbolički jezici visoke razine

```
#include<iostream>
using namespace std;
int main()
{
    int brojac,N;
    cout<<"Program ispisuje djelitelje odabranog \
prirodnog broja." <<endl;
upis:cout<<"Upisi prirodni broj: ";
    cin>>N;
    if (N<=0)
    {
        goto upis;
    }
    cout<<"Djelitelji broja "<
    for (brojac=1;brojac<=N;br
    {
        if(N%brojac==0)
            cout<<brojac<<"   "
    }
    cout<<endl;
    return 0;
}
```

```
130 RANDOMIZE
140 REM *** NUMBER OF GUESSES
150 LET A=100
160 LET M=7
170 PRINT "INSTRUCTIONS (1=YES
0=NO)?";
180 INPUT Z
190 IF Z=0 THEN 280
```

Simbolički jezici visoke razine

- U drugoj polovini dvadesetog stoljeća nastaju programski jezici FORTRAN, COBOL, BASIC, PASCAL, programski jezik C i mnogi drugi.
- Simbolički jezici visoke razine se mogu podijeliti na jezike opće namjene i jezike prilagođene određenoj vrsti problema.

Izvorni program

- Program napisan simboličkim programskim jezikom (u obliku koji nije strojni) zove se izvorni program (engl. *source code*).
- Izvorne programe treba prevesti u strojni oblik.
- Prevode ih programi koji se nazivaju jezični prevoditelji.

Jezični prevoditelji

- Jezični prevoditelji se međusobno razlikuju složenošću i djelotvornošću, a načelno se mogu podijeliti u dvije skupine:
  - interpreteri (interpretatori, engl. *interpreter*),
  - kompajleri (kompilatori, engl. *compiler*).

Interpreter

- Interpreter svaku naredbu izvornog programa prevodi u strojni oblik u trenutku izvođenja programa.
- Simbolička naredba se prevodi u jednu ili više naredbi strojnog jezika i zatim se izvrši.
- Nakon toga se prevede sljedeća simbolička naredba i izvrši, i tako redom.

Interpreter

- Prevođenje naredbu po naredbu omogućava trenutno otkrivanje određene vrste pogrešaka i interaktivno ispravljanje.
- Nedostaci su relativno sporiji rad i nužnost isporuke izvornoga kôda programa korisniku.
- Izvorni program je moguće izvršiti samo ako je na računalu prisutan i interpreter.

Kompajler

- Kompajler prevodi izvorni program tako da analizira i prevede cjelokupni izvorni program odjednom.
- Kao rezultat tog rada nastaje izvršni tj. strojni oblik programa.
- Za razliku od interpretera, kod kompajlera su izvorni program i izvršni program potpuno odvojeni i pri izvođenju neovisni.

Kompajler

- Izvršni program se može izvršavati bez postojanja izvornog programa.
- Korisniku se najčešće predaje samo izvršna inačica programa.
- Pošto je taj oblik za čovjeka nečitljiv i nerazumljiv (produžetak naziva com ili exe) na stanovit se način tako štiti trud programera od neovlaštenih prepravaka ili krađe dijelova programa.

Računalne komponente

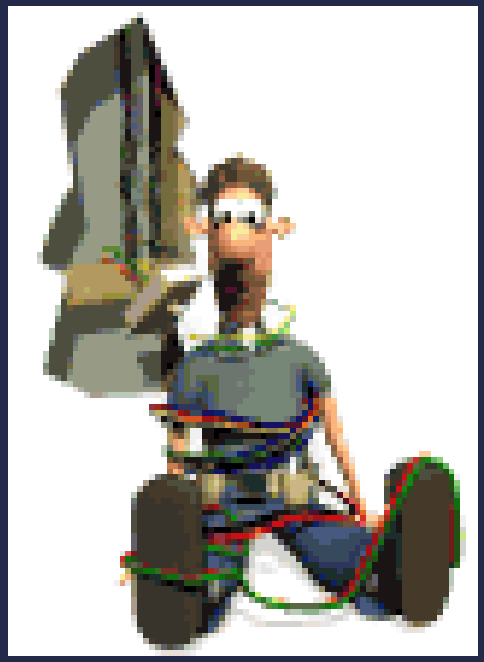
- Računalo se sastoji od mnogo komponenti.

Računalne komponente

- Pouzdanost i djelotvornost računala ovisi o pouzdanosti i djelotvornosti svake od ugrađenih komponenata.

Konstruktor računala

- Konstruktor računala rabi mnogo komponenata koje spaja u cjelinu.
- Kada bi se morao baviti unutarnjom građom, kvalitetom i pouzdanošću svake od komponenti projektiranje računala bilo bi vrlo otežano.



Proizvođači komponenti

- Umjesto konstruktora računala, za unutarnju građu i pouzdanost svake komponente brine se njen proizvođač.
- Proizvođač pojedine komponente može se tehnološki specijalizirati za njezinu proizvodnju te tako proizvoditi komponente visoke kvalitete i pouzdanosti.
- Npr. postoje proizvođači koji su se specijalizirali za proizvodnju tvrdih diskova, tipkovnica, monitora itd.

Proizvođači komponenti

- Komponente su normirane pa se mogu ugraditi u različite elektroničke uređaje što omogućava proizvodnju velikih serija i sniženje cijena.

Konstruktor računala

- Konstruktor rabi gotove komponente i ne mora poznavati njihovu unutarnju građu.
- Dovoljno je da zna zadaću koju komponenta obavlja i način njezina spajanja s ostalim dijelovima računala.
- To mu bitno olakšava posao i omogućuje izradu pouzdanijih i djelotvornijih računala.

Objekti

- Zamisao uporabe gotovih komponenti pri gradnji složenih sustava primjenjena je i pri izradi programa
- Zadatak se dijeli na manje dijelove koji se mogu neovisno rješavati i provjeravati.
- Gotovi dijelovi programa (komponente) nazivaju se objekti (engl. *object*).

Objektno orijentirani program

- Programi koji rabe objekte nazivaju se objektno orijentirani programi (engl. *OOP, object oriented programs*).
- Objekti se mogu pisati i provjeravati odvojeno od cjeline i rabiti u različitim programima.

Prenosivi programski jezici

- Prenosivi (engl. *portable*) programski jezici ne ovise o sklopovlju i operacijskom sustavu.
- To, primjerice, znači da se isti program može, bez izmjena, izvoditi na različitim računalima koja rade pod različitim operacijskim sustavima.

Prenosivi programski jezici

- S pojavom Interneta raste potreba za prenosivosti.
- Razvijaju se novi programski jezici od kojih je najpoznatiji Java.
- Java je snažan objektno orijentirani programski jezik opće namjene.

Java prevoditelji

- Prenosivost je riješena pomoću dva programa prevoditelja:
  - Java kompajler (program koji se mora nalaziti na računalu na kojem programer piše i prevodi izvorni program),
  - Java prividno računalo (program koji se mora nalaziti na računalu na kojem se program želi izvršiti).

Java kompajler

- Programer piše izvorni program naredbama programskog jezika Java.
- Zatim se izvorni program prevodi Java kompajlerom.
- Prevođenjem nastaje program *Java bytecodes* što je međukorak do konačnog strojnog oblika programa.
- *Java bytecodes* je oblik programa koji se ne može izravno izvršiti niti na jednom računalu ali se može proslijediti u istom obliku svakom računalu bez obzira na vrstu i operacijski sustav.

Java bytecodes

- *Java bytecodes* je potpuno prenosiv oblik programa, dakle prihvatljiv svakom računalu.
- To nije strojni oblik programa ni za jedno stvarno računalo već je "strojni oblik" za Java prividno ili virtualno računalo (engl. *Java virtual machine*, *Java engine*).

Java prividno računalo

- Java prividno računalo (Java virtualno računalo) je računalni program prevoditelj koji prevodi *Java bytecodes* u strojni jezik računala na kojem se program izvršava.
- Rezultat java prividnog računala je strojni jezik koji računlao ”razumije” i može ga izvršiti.
- Za svaki procesor mora postojati posebno Java prividno računalo (program prevoditelj) koji će *Java bytecodes* pretvoriti u strojni jezik tog procesora.

Java program