ALGORITMI I PROGRAMIRANJE RAČUNALA

4 sata

SADRŽAJ PREDAVANJA

ALGORITMI **2,5 sati**

Operativni ciljevi

- **Upoznati se s algoritmima, njihovom formom, radom, svrhovitošću, te uočiti vezu između algoritama i programiranja**

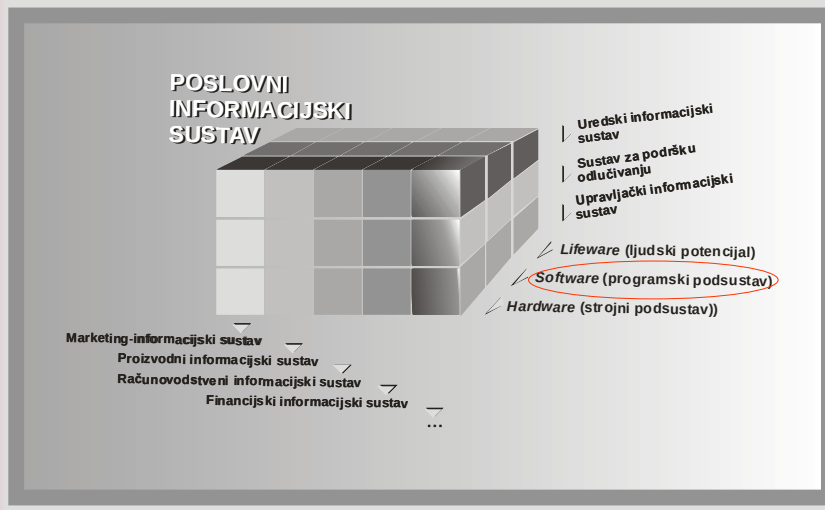
PROGRAMIRANJE I PROGRAMSKI JEZICI **1,5 sati**

Operativni ciljevi

- **Upoznati se s programiranjem, vezom stroj-program, programskim jezicima, generacijama, evolucijom programskih jezika, te trendovima u razvitku programskih jezika**

ALGORITMI

POSLOVNI INFORMACIJSKI SUSTAVI



POJAM ALGORITAM

- **Za pojam zaslužan arapski matematičar iz IX. stoljeća: Muhamed ibn Musa Al Horezmi (u prijevodu: *Muhamed sin Muse iz Horezma*)**
- **Napisao knjigu u kojoj je razradio postupke i pravila za provođenje aritmetičkih operacija s brojevima zapisanim u dekadskom obliku.**

Al Gore

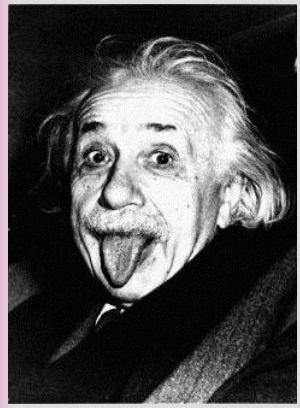
PRIMJER RAČUNSKOG ALGORITMA

- **U ručnoj (manualnoj) obradi podataka, skup postupaka koje treba učiniti da bi se riješio zadatak**
- **npr. Algoritam je i naučen postupak množena u osnovnoj školi**

```
 23* 32
 ----
 69
  46   +
 ----
 736
```

TKO ŠTO PROCESUIRA?!*

Algoritam **Program**



* proces od *lat.* *processus* – napredak, rastanje

ELEKTRONIČKA OBRADA PODATAKA

Obrada podataka koja se pretežito obavlja na elektroničkom računalu, prema unaprijed zadanom programu (nizu instrukcija (naredbi)) i nad podacima koji su upisani i nalaze se u centralnoj memoriji.

EVOLUTIVNI NIZ

- **Manualna obrada**
- **Mehanička obrada**
- **Elektromehanička obrada**
- **Elektronička obrada**

PREDMET OBRADE

Podaci (sve što se može digitalizirati: brojevi, alfanumerički znakovi, slike, video, tonski zapisi)

FAZE NASTANKA PROGRAMA

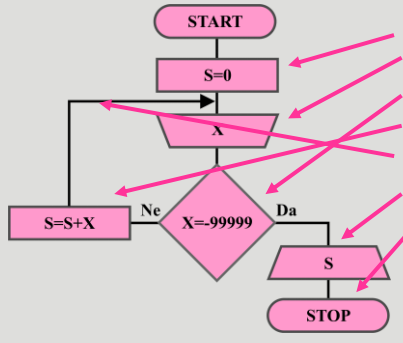
Algoritmi - faza u procesu izrade programa za elektroničku obradu podataka

- **Definiranje problema**
- **Razumijevanje problema**
- **Izrada algoritma (iskusni programeri u glavi)**
- **Prevođenje algoritma u programski jezik**
- **Strojno prevođenje (interpretacija) i testiranje programa**
- **Korekcija i prilagodba programa**
- **Izrada popratne dokumentacije i uputa za rad**

ODNOS ALGORITMA I PROGRAMA

- **S obzirom da su algoritmi priprema za izradu programa, kreiraju se u duhu programskog jezika za koji su priprema.**
- **Prilikom kreiranja algoritama treba težiti da instruktivni odnos**

algoritam : program bude 1:1




```
10 S=0
20 INPUT X
30 IF X=-99999 THEN GOTO 60
40 S=S+X
50 GO TO 20
60 PRINT S
70 END
```

ALGORITMI - KARAKTERISTIKE

Algoritam je:

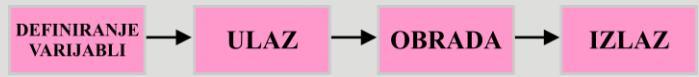
- **Recept u kuharici,**
- **Upute za uporabu,**
- **Proces spajanja gena,**
- **Itd.**

Što su bitne karakteristike algoritma?

- **Orijentiranost k cilju (rezultatu),**
- **Konačnost (upotrebljivost),**
- **Ponovljivost uz iste ulazne uvijete,**
- **Razumljivost procesnoj jedinici,**
- **Instruktivnost (formira se u obliku naredbi)**

NAČELNA STRUKTURA ALGORITMA

Opći algoritmi (zbrajanje unesenih brojeva):



Specijalizirani algoritmi (računanje broja e):

OD ALGORITMA DO IZVRŠNOG KODA

Algoritam se zapisuje u:

1. Obliku pseudo (meta) jezika (govornog jezika koji oponaša programski jezik) i/ili
2. Grafičkom obliku, tzv. Blok dijagram ili dijagram tijeka programa

Program se zapisuje u:

Programskom jeziku

Prevođenje

Program se izvodi u:

Strojnom jeziku

PROCES NASTAJANJA ALGORITMA

- **Definiranje problema**
- **Razumijevanje problema**
  - **Analiza (rastavljanje na sastavne dijelove) problema i "pronalaženje" postupka kojim problem riješiti na stroju razumljiv način (primjer s kutijom šibica)**
- **Pisanje algoritma**
- **Testiranje algoritma (Traženi nivo znanja je testiranje (procesuiranje) algoritma i prepoznavanje svrhovitosti algoritma)**
- **Ispravljanje algoritma (Greške sintakse; formalne i logičke)**
- **Kreiranje dokumentacije**

ALGORITMI
3. Generacije programskih jezika
(Basic)

STRUKTURA ALGORITAMA

Algoritam grade(algoritmi koje ćemo izučavati su algoritmi za obradu numeričkih podataka)

- **Predmeti i sredstva obrade (objekti - podaci):**
  - **Varijable**
  - **Konstante**
  - **Matematički i logički operatori**
- **Algoritamske strukture**

Pravila sintakse

***grč. syntaxis* – sastavljam, sređujem; gram. Skladnja, u ovom slučaju algoritma/programa**

VRSTE PODATAKA (OBJEKATA)

Kompjuterom se obrađuje sve što se "može digitalizirati" – konvertirati u broj (u binarni brojevni sustav – 0 i 1)

Izdvajamo:

- **Numeričke podatke (brojevi)**
  0, 111.001, 10E23, …
- **Alfanumeričke podatke (brojevi i slova)**
  "Pero", "0", "111.001", "OSIJEK", …
- **Ostali tipovi: logički,**
  datumski, memo, …

VARIJABLE

***franc. varijable, mat.* promjenjiva veličina, promjenljiva vrijednost**

Označavaju se simbolima kao i u matematici (postoje za pojedine programske jezike pravila označavanja, tzv. pravila notacije)

- **Primjeri varijabli:**

X, y, z, broj, ime, god, nIznos, …

KONSTANTE

lat. constans mat. **stalan, nepromjenljiva veličina**

Označavaju se simbolima kao i u matematici (postoje za pojedine programske jezike pravila označavanja, tzv. pravila notacije)

Konstante tijekom izvođenja algoritma ne mijenjaju početno pridruženu vrijednost (najpoznatije su e=2,718, Pi=3,141)

- **Primjeri konstanti:**
- **Pi, e, p, g, nGravit, …**

OPERATORI

mat. **znak (predznak) kojim se obilježava način izvođenja operacije**

Označavaju se matematičkim simbolima:

=, +, -, *, /, ^, (,), <, >

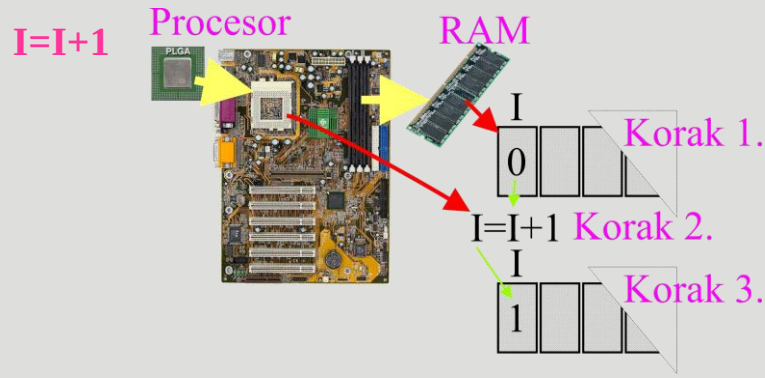
Koriste se u aritmetičkim i logičkim operacijama

Operacije se izvode od lijeva na desno pritom poštujući pravila:

- **Komutacije**
- **Asocijacije**
- **Stupnjevanja**

PARADOKS ZNAKA JEDNAKOSTI

U aritmetičkim operacijama znači pridruživanja (u nekim programskim jezicima se označava s :=)



- **U logičkim operacijama znači vaganje**

Ako je i=10 onda

ALGORITAMSKE STRUKTURE

Slijedna (linearne ili sekvencijalne)

- **Početak i kraj**
- **Definiranje varijabli i konstanti**
- **Ulaz**
- **Izlaz**
- **Aritmetičke i logičke operacije**

Struktura bezuvjetnog skoka

Struktura grananja (sadrži logičke operacije), a kombinira se sa:

- **Slijednom strukturom**
- **Strukturom bezuvjetnog skoka**

Struktura iteracije (ponavljanja ili petlje)

SLIJEDNA STRUKTURA

PSEUDO JEZIK **BLOK DIJAGRAM**

Početak

Definiranje varijabli i konstanti

1. X=0, Y=0, S=0

Ulaz **2. Učitaj x**

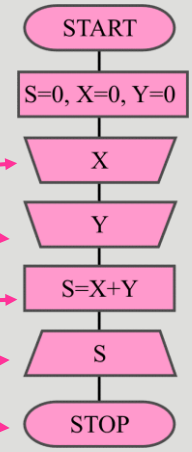
3. Učitaj y

Aritmetička operacija

4. S=X+Y

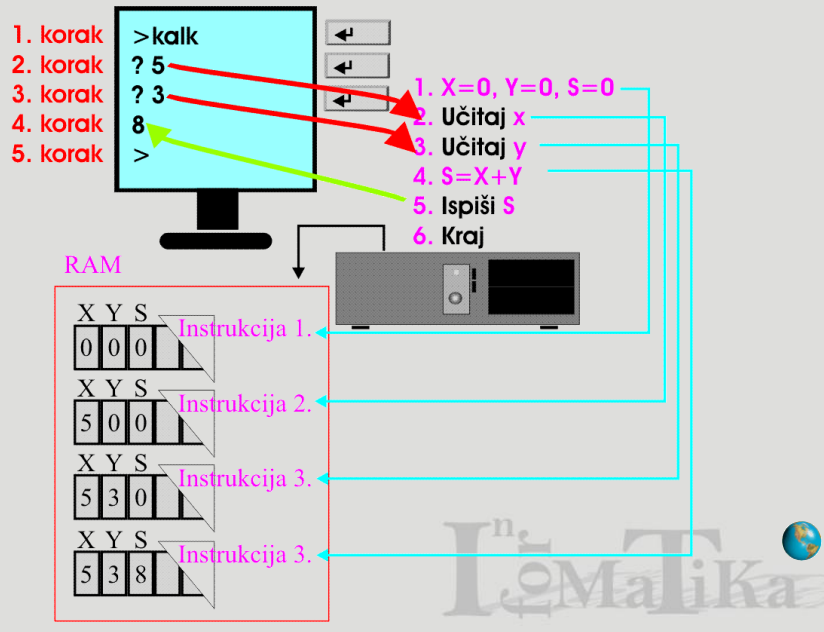
Izlaz **5. Ispiši S**

Kraj **6. Kraj**



Kod pisanja algoritma u pseudo jeziku (kodu) linije se obično označavaju rednim brojevima

PROCES IZVOĐENJA ALGORITMA



TESTIRANJE ALGORITMA

- **Izvodi se kao i u matematici uvrštavanjem vrijednosti u algoritam**
- **Algoritam se testira sekvencialno praćenjem svakog reda (instrukcije) algoritma od početka do kraja, uz zapisivanje vrijednosti koje varijable usput poprimaju, da bi se u konačnici saznala konačna vrijednost izlaznih varijabli**

TESTIRANJE ALGORITMA

Greške u algoritmu (Bug-ovi: *eng. bug* – stjenica, kukac, buba u glavi, kvar, zaraza):

- **Sintaksne greške (*grč. syntaxis* – sastavljam, sređujem; *gram.* Skladnja)**
- **Formalne greške su "pravopisne" greške i "tipfeler"**

npr. 10. Iskiši S **isp.** 10. Ispiši S
23. A-A+1 **isp.** 23. A=A+1

Logičke greške najčešće nastaju zbog:

- **Nerazumijevanja problema**
- **Nepoznavanja logike rada stroja**
- **Niskog programerskog iskustva**

Greške se ispravljaju tzv. debugiranjem

LOGIČKE GREŠKE - slijedna struktura

Nedefinirane varijable:

npr. 1. Učitaj A
2. S=S+1
Koliki je S?

isp. 1. S=0, A=0
2. Učitaj S
3. S=S+1

Dijeljenje s nulom:

npr. 1. A=0, S=0
2. Učitaj B
3. S=B/A
Koliki je S?

- Kako spriječiti unos nulu kao djelitelja?

BEZUVJETNI SKOK

Struktura bezuvjetnog skoka omogućava narušavanje linearnosti

PSEUDO JEZIK

x. Idi na y

BLOK DIJAGRAM

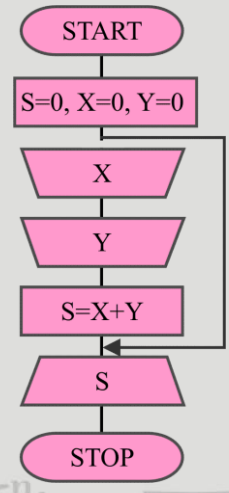
Gdje su x i y brojevi linija algoritma bez obzira na smjer (gore/dolje)

- Koristi se za testiranje algoritma (preskače dio algoritma)
- Izaziva grešku bezuvjetnog ponavljanja (tzv. beskonačna petlja ili iteracija)
- Kombinira se s strukturom grananja radi narušavanja linearnosti / uspostavljanja ponavljanja (dijela) algoritma

BEZUVJETNI SKOK- testiranje algoritma

Omogućava preskakanje dijela algoritma

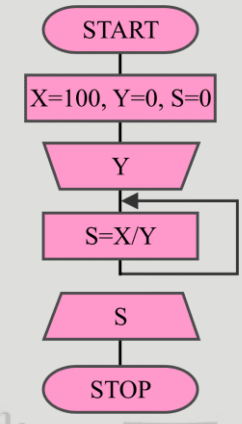
1. X=100, S=0
2. Idi na 5
3. Učitaj Y
4. S=X/Y
5. Ispiši S
6. Kraj



BEZUVJETNI SKOK- beskonačna itracija

Česta greška zbog koje je bezuvjetni skok na lošem glasu

1. X=100, S=0
2. Učitaj Y
3. S=X/Y
4. Idi na 3
5. Ispiši S
6. Kraj



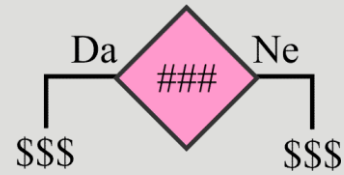
GRANANJE

Struktura grananja koristi se za narušavanje linearnosti

PSEUDO JEZIK

x. Ako je ### onda $$$ [u protivnom $$$]

BLOK DIJAGRAM



Gdje su

- **x** broj linije algoritma
- **###** logička operacija
- **$$$** slijedna operacija i/ili operacija bezuvjetnog skoka (tzv. uvjetni skok)

GRNANJE

LOGIČKE OPERACIJE (###)

Uspoređivanje:

- **varijable s varijablom (x s y, z s k, i sl.)**
- **varijable s brojem (x s 0, y s -9999, i sl.)**

Logički operatori:

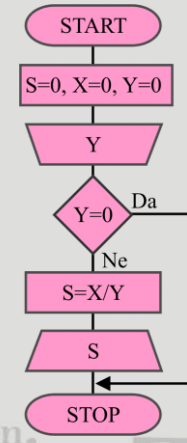
- **=** jednako
- **>** veće od
- **<** manje od
- **>=** veće ili jednako
- **<=** manje ili jednako
- **<>** različito od

GRANANJE

u kombinaciji sa strukturom bezuvjetnog skoka

(spriječavanje unosa nule kao djelitelja)

1. X=100, S=0, Y=0
2. Učitaj Y
3. Ako je Y=0 Idi na 6
4. S=X/Y
5. Ispiši S
6. Kraj

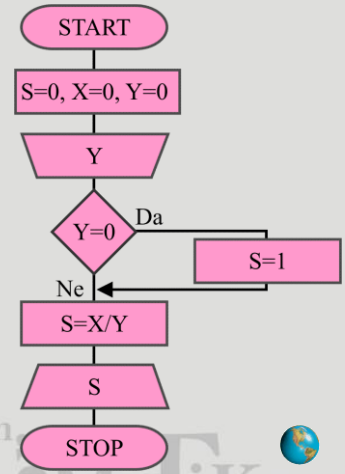


GRANANJE

u kombinaciji sa slijednom strukturom

(spriječavanje unosa nule kao djelitelja?!)

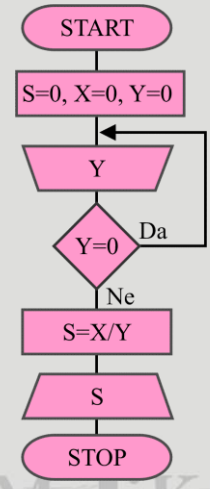
1. X=100, S=0, Y=0
2. Učitaj Y
3. Ako je Y=0 onda Y=1
4. S=X/Y
5. Ispiši S
6. Kraj



GRANANJE

u kombinaciji sa strukturom bezuvjetnog skoka (ponavljanje - spriječavanje unosa nule kao djelitelja)

1. X=100, S=0, Y=0
2. Učitaj Y
3. Ako je Y=0 onda idi na 2
4. S=X/Y
5. Ispiši S
6. Kraj



GRNANJE – LOGIČKA GREŠKA

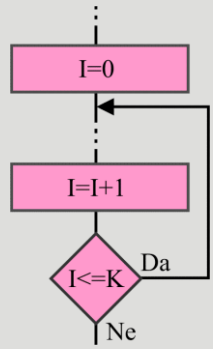
Skok na nepostojeću adresu (kod pseudo koda)

1. X=100, S=0, Y=0
2. Učitaj Y
3. Ako je Y=0 onda idi na 7
4. S=X/Y
5. Ispiši S
6. Kraj

Nastaje zbog ažuriranja redova algoritma bez ažuriranja instrukcija koje sadrže skok (savjet: numerirati svaki red s korakom 10 – npr. 100, 110, 120….)

ITERACIJA (PONAVLJANJE)

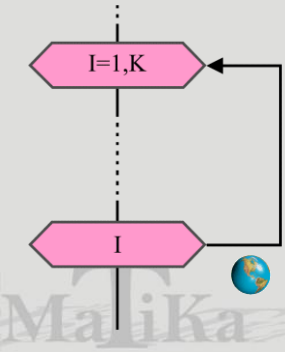
- **Evoluirala iz kombinacije slijedne strukture i strukture uvjetnog skoka**
- **Više vrsta iteracija - bavit ćemo se samo iteracijama s unaprijed definiranim konačnim brojem koraka**



x. Za I=1 do K

……

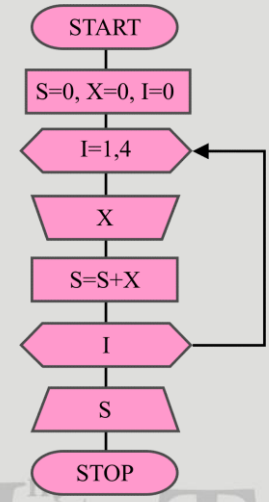
z. Povećaj I



ITERACIJA (PONAVLJANJE)

Omogućava učitavanje konačnog broja brojeva, te računa kumulativ

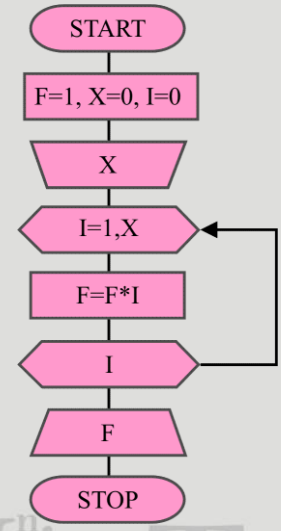
1. **S=0, X=0, I=0**
2. **Za I=1 do 4**
3. **Učitaj X**
4. **S=S+X**
5. **Povećaj I**
6. **Ispiši S**
7. **Kraj**



ITERACIJA (PONAVLJANJE)

Izračun Faktorijela (X!)

1. F=1, X=0, I=0
2. Učitaj X
3. Za I=1 do X
4. F=F*I
5. Povećaj I
6. Ispiši F
7. Kraj



PITANJA!

PROGRAMIRANJE

TEMELJNI POJMOVI

- **PROGRAMIRANJE**
  postupak kreiranja programa
- **PROGRAM**
  skup instrukcija (uputa) računalu kako da radi
- **PROGRAMSKI JEZK**
  skup pravila* (sintaksa) za pisanje programa
- **STROJNI JEZIK**
  instrukcije "ugrađene" u procesor temeljem koji procesor manipulira s podacima

*** Zapisan algoritam prema tim pravilima i "pohranjen u računalu" u obliku teksta posebnim programima moguće je prevesti u stroju razumljiv kod (strojni jezik)**

RAZINE PROGRAMSKIH JEZIKA

- **Niži programski jezici**

  Zahtijevaju dobro poznavanje strukture računala, procesorske instrukcije i procese u računalu – poznavanje logike rada stroja

  Primjer: Asembler – simbolički jezik

- **Viši programski jezici**

  Nastali objedinjivanjem više asemblerskih instrukcija u jednu, stoga bliži načinu razmišljanja čovjeka, te se programer može usredotočiti na problem. Što je programski jezik na višem nivou to je manje potrebno znati o principima rad računala (vidi generacije)

  Primjer: Basic, Pascal, Fortran,….

PRIMJER ASEMBLERSKOG PROGRAMA

- Prikazani kod programa napisan je u programskom jeziku…..

LD	**A,2**	U registar A upiši broj 2
LD	**B,3**	U registar B upiši broj 3
ADD	**A,B**	U registar A upiši vrijednost registra A uvećanog za registar B

10 bodova

GENERACIJE PROGRAMSKIH JEZIKA

- **I. generacija**
  Strojni kod
- **II. generacija**
  Asembleri (jedna instrukcija strojnog koda = jedna asembler instrukcija)
- **III. generacija**
  Nastala objedinjavanjem više asemblerskih instrukcija u jednu (Basic, Fortran, Cobol, Pascal, PL1, C, …)
- **IV. generacija**
  Ne proceduralni jezici (upitni jezici, generatori izvješća, generatori aplikacija)
- **V. generacija**
  Prirodni govorni jezik

NAMJENA PROGRAMSKIH JEZIKA

- **Nekada:** **Matematički**
  Primjer: Fortran (Formula+Translator)
  Poslovni
  Primjer: Cobol
  Konceptualni
  Primjer: C
  Orijentirani na baze podataka
  Primjer: SQL
  Univerzalni
  Primjer: PL1
  ••••••••
- **Danas:** Svi programski jezici teže univerzalnosti (i sa stajališta namjene i sa stajališta platforme)

ORIJENTACIJA PROGRAMSKIH JEZIKA

- **Jezici orijentirani ka sistemskom* programiranju**
  Zahtijeva dobro poznavanje građe stroja i procesa koji se u stroju odvijaju
- **Jezici orijentirani ka korisničkom programiranju**
  Zahtijeva dobro poznavanje sustava obrade podataka za koji se piše program
- **Kombinirani jezici**

* Programiranje za potrebe operacijskog sustava (programa koji omogućava elementarnu komunikaciju između stroja i čovjeka)

TIPOVI PROGRAMIRANJA

- Jednostavno programiranje
- Strukturirano programiranje (uvođenje reda u jednostavno programiranje)
- Modularno programiranje (moduli po načelu crne kutije)
- Objektno programiranje (objektu se mijenjaju svojstva)
- Makro programiranje (integracija izvodljivih programa – procesa)
- Markup programiranje (HTML)

. . . . . .

TIPOVI PROGRAMIRANJA

- Izrada Algoritma: (Definiranje problema, Razumijevanje problema, Pisanje algoritma, Testiranje algoritma, Ispravljanje algoritma, Kreiranje dokumentacije)

Prevođenje Algoritma u programski jezik

Zapisivanje programa u računalu

Prevođenje programa u strojni jezik

Testiranje programa

Ispravljanje grešaka (BUG-ovi)

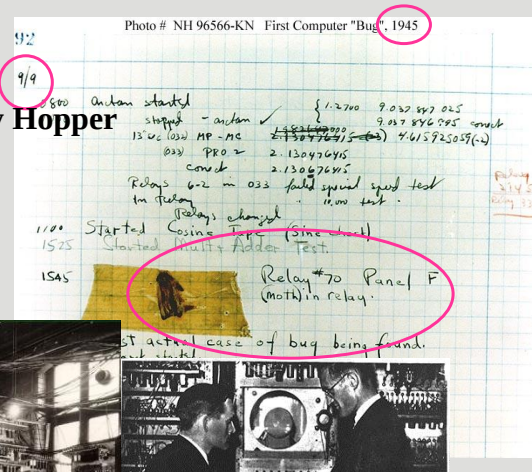
} **Iterativni proces**

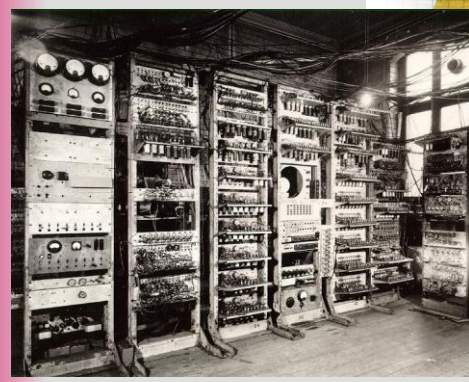
Izrada dokumentacije (opisa programa i upute za rad)

- Kvalitetna priprema (izrada algoritma) ključ dobrog programiranja

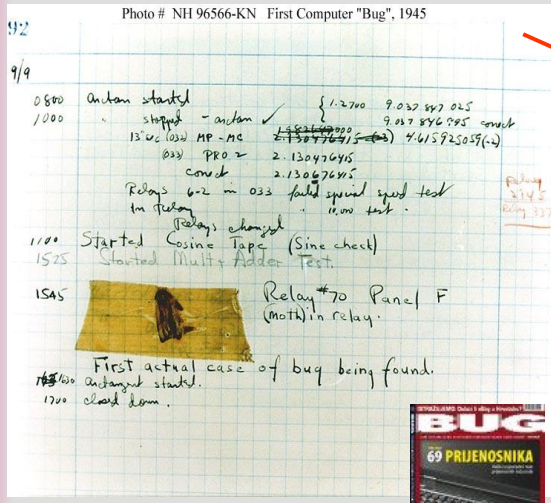
BUG-ovi

- Zapisala **Grace Murray Hopper**
- Harvard University
- Računalo Mark II Aiken Relay Calculator (a primitive computer).





PRIČA O BUG-u



ALGORITAM2PROGRAMSKI JEZIK

Algoritam	Basic	Struktura
UČITAJ	INPUT	Slijedna
ISPIŠI	PRINT	
KRAJ	END	
IDI NA	GOTO	Bezuvjetni skok
AKO JE ONDA U PROTIVNOM	IF THEN ELSE	Uvjet (grananje)
ZA DO	FOR TO	Iteracija
POVEĆAJ	NEXT	

PROGRAMSKI JEZIK2STROJNI JEZIK

- **Program napisan u programskom jeziku (tekst)**

  Interpreter (simultano prevođenje) **Compiler**

- **Program na strojnom jeziku (u binarnom obliku zapisane instrukcije koje razumije procesor)**

  Dok se instrukcije prevode one se i izvršavaju (program radi), tako da rezultat interpretiranja nije binarna datoteka

  Prevedeni oblik pohranjuje se i čuva u binarnom obliku (datoteci) i po potrebi na zahtjev korisnika pokreće i izvršava (datoteke tipa EXE i COM)

- **Načelno:**
  Interpretiranjem se brže pokreće program, ali on sporije radi!

IZVORNI (NEPREVEDENI) PROGRAMI

- Tekstovi koji se pohranjuju u datoteke na uređajima vanjske memorije (diskove, diskete,…)
- Obično su te datoteke označene posebnim nastavcima koji aludiraju na programski jezik u kojem je program napisan
- Primjer naziva datoteke:

  ime.nastavak (nastavak 3 slova)

  BAS-Basic, PAS-Pascal, FOR-Fortran

PROGRAMI PREVODITELJI

- Binarni program (kao datoteka smješten na uređaju vanjske memorije) koji nakon startanja simultano čita datoteku u kojoj je smješten izvorni program, prevodi instrukciju po instrukciju u procesoru razumljiv niz instrukcija strojnog koda,
- **(ako je interpreter)** a procesor te instrukcije izvršava i tako izvodi program
- **(ako je compiler)** a rezultat prevođenja smješta kao datoteku (binarnu) na uređaj vanjske memorije. Ovakva datoteka izvodljivi je program.

Uređaji u računalu nosioci programa:

- **ROM čipovi**
  primaju jednostavne programe u strojnom jeziku i omogućavaju osnovne "životne" funkcije stroja
- **Vanjska memorija**
  čuva programe u izvornom (programskom) jeziku u tekstualnom obliku i programe u prevedenom - strojnom (binarnom) obliku kao datoteke
- **RAM**
  preuzima programe na zahtjev procesora (zahtjev nastaje temeljem izvođenja nekog drugog programa, npr. programa operacijskog sustava) sa vanjske memorije i instrukcije u strojnom obliku predaje procesoru koji ih izvršava

Što će se od programiranja nadalje raditi:

- **SISTEMSKI SOFTVARE** (programi)

  BIOS, operacijski sustavi, pomoćni sistemski programi, programi prevodioci, …

- **KORISNIČKI SOFTVARE** (programi)

  Poslovni uredski programi, programi za upravljanje s bazama podataka, programi za pomoć u odlučivanju, …

Za sve što se u stroju dešava ili ne dešava odgovorni su PROGRAMI!

PITANJA!