Elektrotehniško-računalniška strokovna šola in gimnazija Ljubljana

Melita Kompolšek

# OSNOVE ITK

Učbenik za 1. letnik srednjega strokovnega izobraževanja

TEHNIK RAČUNALNIŠTVA

Elektrotehniško-računalniška strokovna šola in gimnazija Ljubljana

Melita Kompolšek

# Osnove ITK

Učbenik za 1. letnik srednjega strokovnega izobraževanja

TEHNIK RAČUNALNIŠTVA

Melita Kompolšek, Elektrotehniško-računalniška strokovna šola in gimnazija Ljubljana

# Osnove ITK

Učbenik za 1. letnik srednjega strokovnega izobraževanja TEHNIK RAČUNALNIŠTVA

Strokovni recenzenti:

dr. Uroš Breskvar, Elektrotehniško-računalniška strokovna šola in gimnazija Ljubljana

doc. dr. Samo Simončič, Fakulteta za strojništvo, Univerza v Ljubljani

Aleš Volčini, Elektrotehniško-računalniška strokovna šola in gimnazija Ljubljana

Jezikovna recenzentka:

Marjeta Šušteršič Menart, Elektrotehniško-računalniška strokovna šola in gimnazija Ljubljana



CIP - Kataložni zapis o publikaciji
Narodna in univerzitetna knjižnica, Ljubljana

659.2:004(075.3)(0.034.2)

KOMPOLŠEK, Melita
Osnove ITK [Elektronski vir] : učbenik za 1. letnik srednjega strokovnega izobraževanja Tehnik računalništva / Melita Kompolšek. - El. knjiga. - Ljubljana : Elektrotehniško-računalniška strokovna šola in gimnazija, 2016

Način dostopa (URL):
: http://moodle.vegova.si/pluginfile.php/38527/mod_resource/content/1/Osnove%20ITK.pdf

ISBN 978-961-93617-6-4 (pdf)

286861312

**KAZALO**

# PREDGOVOR

Gradivo Osnove ITK je namenjeno dijakom 1. letnika srednjega strokovnega izobraževanja, smer tehnik računalništva, oziroma vsem začetnikom, ki želijo usvojiti osnovno znanje za delo z računalnikom. Zgradba in vsebina gradiva omogočata bralcu hitro in enostavno dostopanje do želenih informacij, kar ga še dodatno motivira in s tem nehote povečuje njegovo učinkovitost pri pridobivanju in iskanju novih informacij.

Gradivo je namenjeno vsakomur, tudi popolnim začetnikom, ki želijo spoznati ali izpopolniti svoje računalniške veščine. Zato je vsebina razdeljena na logično in smiselno zaključena poglavja, iz katerih je jasno razvidna, njihova tematika. Za preverjanje pridobljenega znanja so dodane naloge in vprašanja iz obravnavanega sklopa, ki smiselno sledijo zaključku vsakega vsebinskega dela. Gradivo obravnava vsebine, navedene v katalogu znanja, pri čemer predznanje ni potrebno, kar pa ne pomeni, da ni zaželeno. Vse, kar potrebujete, je želja po znanju, ostalo je v gradivu, ki je pred vami.

Avtorica

# 1 RAČUNALNIŠTVO IN INFORMATIKA

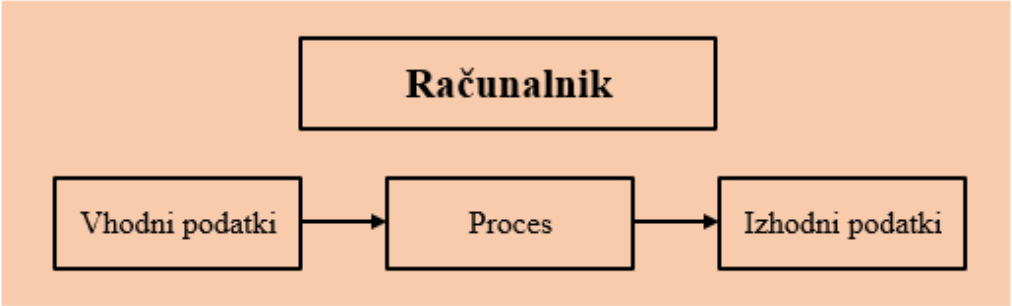
Z razvojem novih tehnologij so postale informacije dostopne vsakomur, vendar pa to od posameznika zahteva poznavanje določenih informacijskih tehnologij. Pri tem ne gre samo za golo tehnologijo (kamor uvrščamo procese, postopke itd.), ampak veliko več, govorimo lahko tudi o napravah, ki podpirajo informacijsko tehnologijo. V to skupino uvrščamo računalnik, ki je dandanes nepogrešljivo orodje za pridobivanje in analizo podatkov oziroma informacij. Zato je smiselno, da v nadaljevanju spoznamo osnovne funkcije in zgradbo računalnika.

## 1.1 Računalnik

### 1.1.1 Funkcije računalnika

Digitalni računalnik opravlja naslednje funkcije:

- s pomočjo vhodnih enot sprejema vhodne podatke,
- shranjuje podatke oziroma navodila v za to namenjenem prostoru-pomnilniku in jih uporabi, kadar je to potrebno,
- obdeluje podatke in jih pretvori v uporabne informacije,
- generira izhodne podatke in
- nadzira vse zgornje korake.




Slika 1: Delovanje računalnika

### 1.1.2 Definicija

**Računalnik** je elektronska naprava za obdelavo podatkov, ki:

- sprejema in shranjuje vhodne podatke,
- obdeluje vhodne podatke in
- generira izhodne podatke v želeni obliki.

### 1.1.3 Prednosti

Prednosti sodobnega računalnika so:

- Visoka hitrost izvajanja nalog:
  - sposobnost obdelave zelo velikih količin podatkov,
  - izvajanje osnovnih operacij je mogoče meriti v mikrosekundah, nanosekundah in celo pikosekundah,
  - obdelavo velikih količin podatkov v zelo kratkih časovnih intervalih.
- Natančnost:
  - izvajanje vseh nalog s 100-odstotno natančnostjo pod pogojem, da so bili dani pravilni vhodni podatki.
- Kapacitete:
  - sposobnost shranjevanja velike količine podatkov,
  - zmožnost shranjevanja več vrst podatkov, kot so slike, video posnetki, besedila, zvok itd.
  - količina shranjenih podatkov je odvisna od kapacitete spomina računalnika,
  - računalnik ima veliko več prostora za shranjevanje podatkov kot človek.
- Skrbnost:
  - v nasprotju s človekom ne zazna monotonosti, utrujenosti in pomanjkanja koncentracije,
  - neprekinjeno delovanje brez napak,
  - zmožnost ponavljajočega dela z enako hitrostjo in natančnostjo.

- Vsestranskost:
  - prilagodljivost pri opravljanju nalog,
  - zmožnost reševanja problemov, povezanih z različnimi področji,
  - zmožnost reševanja kompleksnih znanstvenoraziskovalnih problemov, kakor tudi zmožnost zabavanja z igranjem iger in podobnim.
- Zanesljivost:
  - dolga življenjska doba,
  - enostavno vzdrževanje.
- Avtomatizacija:
  - sposobnost za samodejno opravljanje dane naloge,
  - ko je program shranjen v računalniku, lahko program in navodila nadzorujejo izvajanje programa brez človeške interakcije.
- Manj dela s tiskanimi dokumenti:
  - Računalniška obdelava podatkov v podjetju vodi k zmanjšanju dela s tiskanimi dokumenti in pospeši celotni proces,
  - ni težav z vzdrževanjem velikega števila papirnatih dokumentov.

### 1.1.4 Slabosti

Čeprav ima sodoben računalnik ogromno število prednosti pa ima tudi nekaj slabosti.

- Je brez sposobnosti sklepanja:
  - nima inteligence,
  - uporabnik mora vsa navodila vnesti v računalnik,
  - računalnik ne more sprejeti odločitve.
- Odvisnost:
  - deluje na ukaz uporabnika, zato je v celoti odvisen od človeka.
- Brez občutkov in izkušenj:
  - nezmožnost sodb, ki temeljijo na občutku, okusu, izkušnjah in predznanju.

## 1.2 Vrste računalnikov

Računalnike lahko razvrstimo na štiri večje tipe.

- **Super računalniki** so najzmogljivejši. Gre za računalniški sistem, sposoben obdelati velike količine podatkov v zelo kratkem času. Večina ima večje število mikroprocesorjev, saj z njimi rešujejo zahtevne numerične probleme.
- **Veliki računalniki** imajo velika ohišja (velikost omare ali večji) in so prevladovali do začetka 70. let prejšnjega stoletja; namenjeni so bili paketni obdelavi podatkov in obsežnim operacijam.
- **Miniračunalniki** (strežnik in delovne postaje) so se pojavili v 70. in 80. letih prejšnjega stoletja; so manjši ter manj zmogljivi od velikih računalnikov, vendar tudi bistveno cenejši. Namenjeni so zahtevnejšim nalogam kot mikroračunalniki ali osebni računalniki.
- **Mikroračunalniki** (osebni računalniki) imajo za osrednjo obdelovalno enoto čip – mikroprocesor. Nastali so v 70. letih in so bili predhodniki današnjih osebnih računalnikov.

## 1.3 Informatika in računalništvo kot znanstveni vedi

**Informatika** je veda o zbiranju, oblikovanju, zamenjavi, urejanju in pretvarjanju podatkov in informacij.

**Računalništvo** proučuje predvsem zgradbo in delovanje računalnika, principe programiranja in reševanja problemov. Ukvarja se s podatkovnimi strukturami, z učinkovitim shranjevanjem in dostopom do podatkov, s proučevanjem razvoja modelov za podporo odločanju in učinkovitemu upravljanju. Na računalništvo vplivajo mnoge vede, med njimi predvsem matematika in fizika.

### 1.3.1 Podatek, informacija in znanje

**Podatek** je opredmeteno dejstvo o določeni stvari, s katerim predstavimo informacijo. Znanje, ki ga posredujemo drugim, predstavimo s podatki. Sporočilo je znanje, ki potuje od oddajnika do prejemnika. Ko prejemnik prejme sporočilo, iz njega razbere podatke in nadgradi svoje znanje.

**Informacija** je vsako sporočilo, ki nam pove nekaj novega. Vsak prejemnik si jo razlaga na svoj način.

Primeri podatkov so številke, črke, besede itd. Informacije pridobimo na podlagi podatkov. Na primer iz zaporedja temperatur: 15 °C, 17 °C, 21 °C lahko sklepamo, da temperatura narašča, kar nam predstavlja informacijo. Na podlagi večletnih informacij o naraščanju temperature lahko razvijemo **znanje** – teorijo o globalnem segrevanju.

### 1.3.2 Merjenje količine informacij

Kot vsako drugo količino lahko merimo tudi informacijo. Enota za količino informacije je **bit** (angl. binary digit), kratica za bit je *b*. Bit je najmanjša in nedeljiva enota informacije. 1 bit predstavlja neko informacijo o opazovanem objektu, ki je lahko 1 (da, angl. true) ali 0 (ne, angl. false), ali katerekoli dve drugi medsebojno izključujoči se enako verjetni stanji. Gre za količino informacije, ki jo dobimo kot rezultat poskusa z dvema enakovrednima vnaprej definiranima stanjema (met poštenega kovanca).

Naslednja večja enota je **bajt**, ki predstavlja 8 bitov, nato pa sledi enota kibibajt, pri čemer predpona kibi pomeni $2^{10}$ ali 1024, torej 1 kibibajt predstavlja 1024 bajtov. Kratica predpone kibibajt je KiB. Enote za merjenje informacij so:

| Predpona | Ime | Velikost |
|---|---|---|
| 1 b | bit | 0 ali 1 |
| 1 B | bajt | 8 bitov |
| 1 KiB | kibibajt | $2^{10}B = 1.024\ B = 8.192\ b$ |
| 1 MiB | mebibajt | $2^{20}B = 104.856\ B = 1.024\ KB$ |
| 1 GiB | gibibajt | $2^{30}B = 1.073.741.824\ B = 1024\ MB$ |
| 1 TiB | tebibajt | $2^{40}B = 1024\ GB$ |
| 1 PiB | pebibajt | $2^{50}B$ |

| | | |
|---|---|---|
| 1 EiB | exbibajt | $2^{60}B$ |
| 1 ZiB | zebibajt | $2^{70}B$ |
| 1 YiB | yobibajt | $2^{80}B$ |

## 1.4 Povzetek

V tem poglavju smo spoznali funkcije računalnika, njegove prednosti in slabosti. Seznanili smo se tudi s pomeni pojmov informatika, podatek, znanje in informacija ter definicijo enote za merjenje količine informacij.

## 1.5 Vprašanja in naloge za preverjanje znanja

1. Naštejte funkcije računalnika.
2. Zapišite definicijo računalnika.
3. Naštejte prednosti in slabosti sodobnega računalnika.
4. Katere so vrste računalnikov?
5. Kateri računalnik je najzmogljivejši?
6. Kaj je informatika?
7. Kaj je podatek?
8. Kaj je znanje?
9. Kaj je informacija?
10. Kaj je bit?
11. Kolikšna je vrednost bita?
12. Koliko različnih znakov ali števil lahko predstavimo z enim bajtom?
13. Koliko KB (kilobajtov) je 1 MB (mega) bajt?

# 2 ZGRADBA RAČUNALNIŠKEGA SISTEMA

Računalnik je naprava, ki omogoča avtomatsko obdelavo podatkov. Te vnesemo v računalnik s pomočjo vhodnih naprav, rezultat obdelave podatkov računalnika pa vidimo s pomočjo izhodnih naprav.

Osnovna delitev računalniške opreme na:

- strojno opremo in
- programsko opremo.

## 2.1 Strojna oprema

Strojno opremo sestavljajo:

- vhodne enote,
- centralna procesna enota,
- vodila in
- izhodne enote.

### 2.1.1 Vhodne enote

**Vhodne enote** so naprave, ki omogočajo vnos podatkov v računalnik.

Vhodne enote so:

- tipkovnica,
- miška,
- igralna palica,
- optični bralnik,
- grafična tablica,
- digitalni fotoaparat,
- digitalna kamera,
- mikrofon …

Kadar vhodni napravi dodamo še mehanizem, ki omogoča povratno informacijo, takšna naprava postane vhodno-izhodna. Ker pa je njen primarni namen še vedno vnos podatkov v

računalnik, jo še vedno obravnavamo kot vhodno napravo. Takšen primer je tipkovnica opremljena z zaslončki pod tipkami, ki kažejo funkcijo tipke v danem programu (tipkovnica Optimus Maximus oblikovalca studia Art. Lebedev).

Slika 2: Tipkovnica

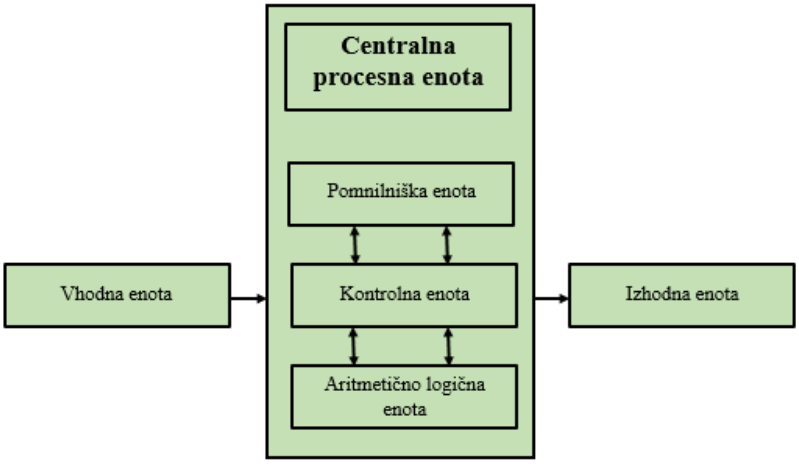
### 2.1.2 Centralna procesna enota

**Centralna procesna enota** je glavni del računalnika, ki obdeluje podatke ter nadzoruje in upravlja ostale enote. Sestavljena je iz digitalnih elektronskih vezij, nameščenih na skupnem integriranem vezju, ki ga imenujemo mikroprocesor. Naloge centralne procesne enote so izvajanje programov, obdelovanje podatkov in kontrola delovanja vhodno/izhodnih naprav. Hitrost izvajanja operacij v centralni procesni enoti merimo v številu operacij na sekundo (npr. MIPS).

Slika 3: Centralna procesna enota

Centralna procesna enota vsebuje tri komponente:

- registre,
- kontrolno enoto in
- aritmetično logično enoto.




Slika 4: Zgradba centralne procesne enote po von Neumannovem modelu

#### 2.1.2.1 Registri

**Register** pomeni eno ali več povezanih pomnilniških celic, v katere je mogoče shraniti neko vrednost. Glavni namen registrov je shranjevanje operandov centralne procesne enote, kar velja predvsem za programsko nedostopne registre. Obstajajo pa še programsko dostopni registri, ki omogočajo delovanje centralne procesne enote. Njuna podvrsta so podatkovno dostopni, kot na primer akumulator pri Motoroli. Registre najdemo v centralni procesni enoti, saj so njen sestavni del.

#### 2.1.2.2 Kontrolna enota

**Kontrolna enota** nadzoruje delovanje vseh delov računalnika, vendar ne obdeluje podatkov.

Funkcije kontrolne enote so:

- odgovornost za nadzor prenosa podatkov in navodil med drugimi enotami računalnika,
- upravljanje in usklajevanje vseh enot računalnika,
- pridobivanje in interpretiranje navodil iz pomnilnika ter usmerjanje delovanja računalnika,
- komunikacija z vhodnimi in izhodnimi napravami za prenos podatkov ali rezultatov iz pomnilnika,
- ne obdeluje ali shranjuje podatkov.

#### 2.1.2.3 Aritmetično logična enota

**Aritmetično logična enota** je sestavljena iz dveh podenot:

- aritmetične enote in
- logične enote.

Funkcija aritmetične enote je opravljanje aritmetičnih operacij, kot so seštevanje, odštevanje, množenje in deljenje. Vse kompleksne operacije opravi tako, da ponavljajoče uporablja zgoraj navedene operacije.

### 2.1.3 Izhodne enote

**Izhodne enote** so naprave, ki omogočajo prikaz podatkov iz računalnika.

Izhodne enote so:

- monitor,
- projektor,
- tiskalnik,
- večnamenske naprave,
- zvočniki …

Kadar izhodni napravi dodamo še mehanizem, ki omogoča vnos informacij, takšna naprava postane vhodno-izhodna. Ker pa je njen primarni namen še vedno prikaz podatkov iz računalnika, jo še vedno obravnavamo kot izhodno napravo. Takšen primer je npr. tiskalnik z detekcijo velikosti papirja.

Slika 5: Monitor

## 2.2 Programska oprema

Programsko opremo delimo na:

- sistemsko programsko opremo – BIOS, operacijski sistemi, gonilniki, sistemska orodja in
- uporabniško programsko opremo – programi za delo z besedili, s preglednicami, za predstavitve, obdelavo slik, grafični programi, programi za delo z zvokom, komunikacijski programi, programi za delo s podatkovnimi bazami, pregledovalniki dokumentov, spletni brskalniki, programska in razvojna orodja, računalniško odprto načrtovanje.

## 2.3 Povzetek

V današnjem času so postali računalniki nenadomestljivi. Uporabljamo jih za reševanje kompleksnih problemov na delovnem mestu, z njihovo pomočjo se lahko tudi zabavamo in komuniciramo. Čeprav so računalniki tako razširjeni, pa le malo ljudi ve, kako delujejo. V tem poglavju smo se seznanili z zgradbo računalniškega sistema, vhodnimi enotami, centralno procesno enoto in izhodnimi enotami.

## 2.4 Vprašanja in naloge za preverjanje znanja

1. Kaj je računalnik in katere so osnovne komponente računalnika?
2. Katere komponente obsega strojna oprema?
3. Ali je monitor izhodna enota?
4. Naštejte pet vhodnih in pet izhodnih enot.
5. Kakšna je razlika med ROM in RAM?
6. V kateri enoti merimo hitrost izvajanja operacij centralne procesne enote?
7. Katero napravo bi uporabili za izhod glasbene datoteke?

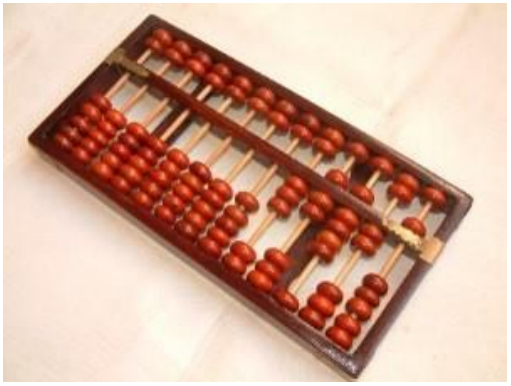
# 3 ZGODOVINA RAČUNALNIŠTVA

Prvi zametki računalništva segajo v čas, ko je bilo štetje in računanje šele v razvoju. V tem času si je človek števila predstavljal s pomočjo prstov na rokah, palic, kamenčkov, narisanih znakov itd.

Slika 6: Štetje s pomočjo prstov na rokah

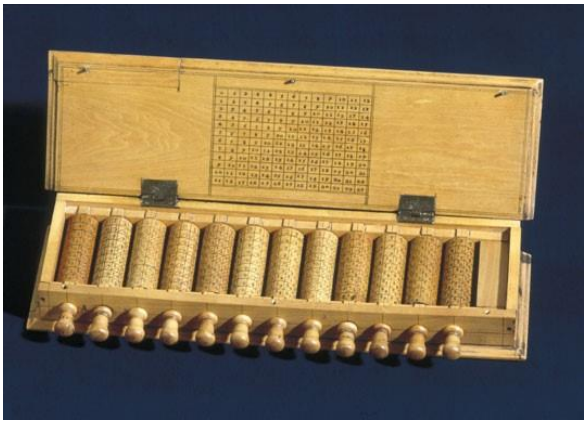
## 3.1 Računala

Najzgodnejše znano računalo je **abak**, ki so ga verjetno razvili Babilonci v obdobju od 2700 do 2300 let pred našim štetjem. Abak je preprost računski pripomoček s kroglicami in služi kot nekakšen pomnilnik za števila, kar omogoča hitrejše računanje.



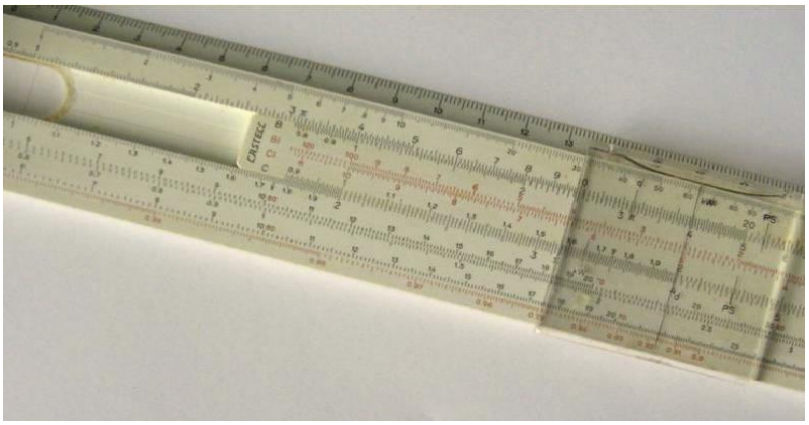
Slika 7: Abakus
Vir: https://hr.wikipedia.org/wiki/Abak (14. 8. 2015)

Leta 1614 je škotski matematik John Napier predstavil definicijo logaritmov in izdelal **prve računske tablice**. Omogočale somnoženje, deljenje, izračun kvadratnega in kubičnega korena.



Slika 8: Prve računske tablice
Vir: www.imgbuddy.com (14. 8. 2015)

Leta 1622 je William Oughtred uporabil Napierjeve logaritme pri izumu **logaritemskega računala**. Najpreprostejše logaritmično računalo omogoča s pomočjo dveh logaritmičnih lestvic množenje in deljenje, računski operaciji, ki sta pri računanju na pamet časovno zahtevni in podvrženi napakam. Uporabnik določi lego decimalne vejice v rezultatu na podlagi miselne ocene. Pri računih, ki zahtevajo tudi seštevanje in odštevanje logaritmov, pa se ti dve matematični operaciji izvedeta posamezno s pomočjo logaritemskega računala, njun rezultat pa je potrebno sešteti oziroma odšteti.



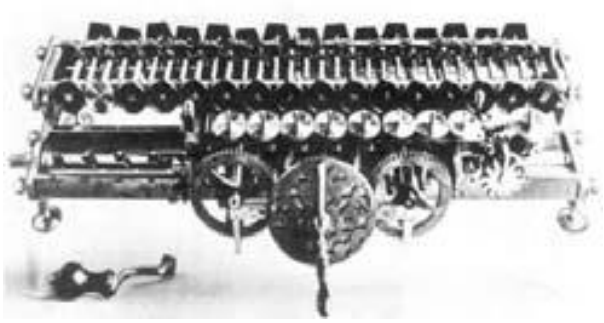
Slika 9: Logaritemsko računalo
Vir: https://sl.wikipedia.org/wiki/Logaritemsko_ra%C4%8Dunalo (14. 8. 2015)

Leta 1642 je Blaise Pascal izumil mehansko računalo **pascaline**, zmožno neposrednega seštevanja in odštevanja, ki ga imamo za začetnika mehanskih računal. Naprava je bila sestavljena iz množice zobnikov s številčnicami na sprednji ploskvi. Uporabnik je vnesel število tako, da je vstavil konico med ustrezni špici na številčnici in jo zavrtel do kovinske zapore. V okencu nad njim se je prikazala števka od 0 do 9. Postopek je nato ponovil za preostanek števila (od desne proti levi). Sešteval je enostavno tako, da je na enak način

vnesel drugo število. Ker so se zobniki lahko vrteli le v eno smer, je odštevanje potekalo po metodi komplementov.

Slika 10: Pascaline
Vir: http://www.ami19.org/Pascaline/IndexPascaline-English.html (14. 8. 2015)

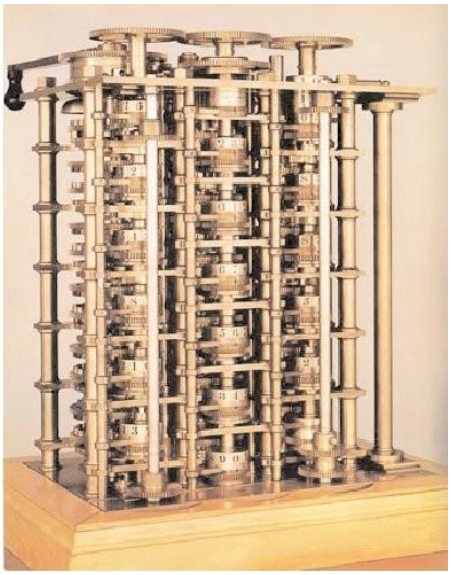
Leta 1672 je Gottfried Wilhelm Leibniz izumil napravo, ki sešteva, odšteva, množi, deli in koreni. Vsebovala je še dodatni dve skupini koles za predstavitev faktorjev ter deljenca in delitelja. Za množenje in deljenje je Leibniz izumil element, imenovan **Leibnizevo kolo**.



Slika 11: Leibnizevo računalo
Vir: http://egradivo.ecnm.si/KIT/mehanska_raunala__kalkulatorji.html (14. 8. 2015)

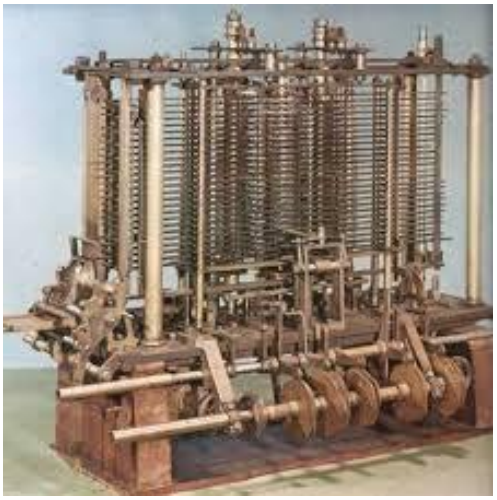
Leta 1837 je Charles Babbage je izumil stroj, ki velja za zasnovo za sodobnega računalnika. Zasnoval je dva tipa strojev:

- **Diferenčni stroj**, ki je bil namenjen za računanje in avtomatično tiskanje matematičnih tabel. Bil je prvi programirani stroj, saj je omogočal strojno izvajanje zaporedij operacij.



Slika 12: Diferenčni stroj
Vir: http://www.educa.fmf.uni-lj.si/izodel/sola/2002/di/bozic/PC_history/do1900.html (14. 8. 2015)

- **Analitični stroj** je vseboval mlin, v katerem so se izvajale operacije, ter pomnilnik za shranjevanje operandov in rezultatov operacij.

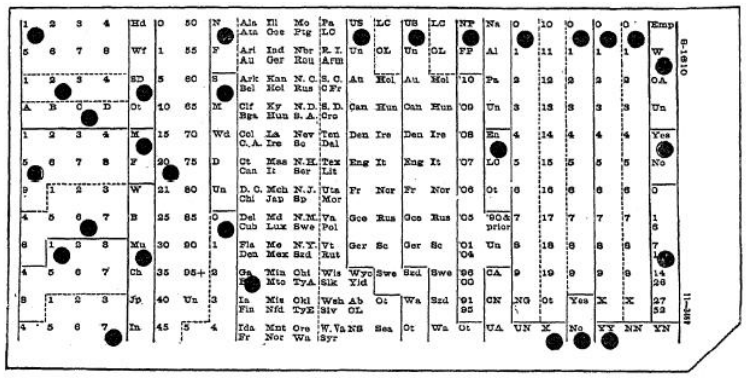


Slika 13: Analitični stroj
Vir: http://www.s-sers.mb.edus.si/gradiva/w3/sistemi/mehano.html (14. 8. 2015)

## 3.2 Elektromehanski stroji

Leta 1890 je Herman Hollerith na predlog ameriške vlade izumil napravo za potrebe popisa prebivalstva. Delovala je na principu **luknjanih kartic**, in sicer tako da se je na vsakem popisnem lističu, odvisno od odgovora (možna sta bila dva) na ustreznem mestu naredila

luknja. Lističe so nato obdelali na napravi, ki je imela kovinske konice za tipanje. Če je konica prišla skozi luknjo v kad z živim srebrom, se je sklenil električni tok.



Slika 14: Luknjana kartica
Vir: https://en.wikipedia.org/wiki/Punched_card (14. 8. 2015)

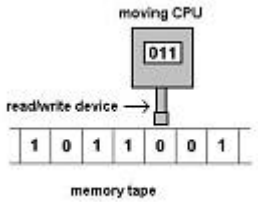
Konrad Zuse je leta 1936 izdelal **binarni električni mehanski računalnik Z1**. Imel je omejene možnosti programiranja in ločene vhodne in izhodne enote. Program je bil zapisan na papirju. Zaradi mehanskih prenosov je bil počasen, zato je izdelal računalnik Z2, ki je že uporabljal releje kot prekinjevalne elemente.

Slika 15: Binarni električni mehanski računalnik Z1
Vir: http://www.nauk.si/materials/1357/out/ (14. 8. 2015)

Alan M. Turing je istega leta ustvaril tako imenovani **Turingov stroj**, ki lahko izračuna vse, kar je mogoče izračunati, pri čemer ni pomemben čas, ampak koraki. Po tem modelu delujejo vse doslej znane računske naprave. V osnovi gre za algoritemski sistem, miselni stroj (abstraktni model), ki stvarno ne obstaja. Alan M. Turing si ga je zamislil, da bi z njegovo pomočjo matematično opredelil algoritem oziroma mehanski postopek/program. Turingov

stroj le oponaša človeško delo in računa po strogem predpisu. Od računalnika se razlikuje po tem, da razčlenjuje in podaljšuje računski postopek, hkrati pa ga standardizira.

V Turingovem stroju so operacije omejene na branje in zapisovanje znakov na trak ali premikanje vzdolž traku levo ali desno. Trak je označen z majhnimi kvadrati. V vsakem koraku operacije stroj zapiše ali prebere le en kvadrat, ki leži pod bralno in pisalno glavo.

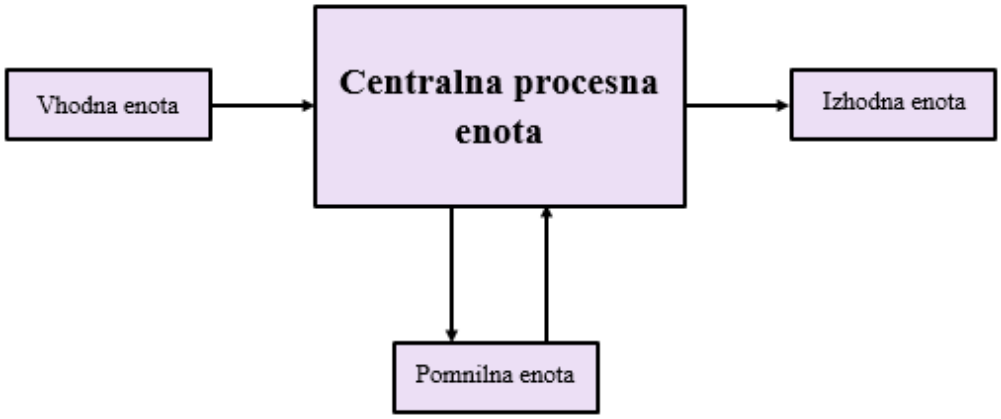



Slika 16: Turingov stroj

Leta 1945 je John von Neumann objavil delo, v katerem je podrobno opisal **princip delovanja računalnika**.

Kot izhodišče je postavil naslednje zahteve:

- Računalnik naj bo splošno uporaben in naj izvaja program avtomatsko.
- Računalnik naj vsebuje centralno procesno enoto in vhodno-izhodne enote. Centralna procesna enota naj bo zgrajena iz aritmetično logične enote, ki bo izvajala operacije računanja, in iz krmilne enote, ki bo razumela ukaze iz pomnilnika ter upravljala računalnik. Preko vhodno-izhodne enote pa bo računalnik izmenjeval podatke med uporabnikom in okolico.
- Ukazi naj bodo shranjeni v enaki obliki in v isti enoti računalnika (pomnilniku) kot podatki, ki jih obdeluje.
- Delovanje računalnika naj bo zasnovano na osnovi dvojiškega številskega sestava. To je potrebno zaradi električne realizacije računalnika, saj električna vezja najlažje ločijo dve diskretni vrednosti. Drugi razlog je logične narave, saj je računalnik namenjen reševanju logičnih problemov, opisanih s stanjema da ali ne.




Slika 17: Von Neumannov model računalnika

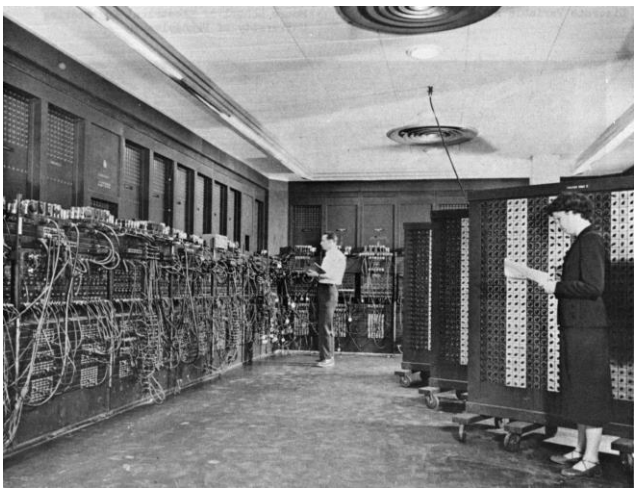
## 3.3 Elektronski računalniki

**Elektronski računalniki** so bili zgrajeni iz elektronskih elementov, in sicer elektronk, relejev in stikal.

### 3.3.1 1. generacija (od 1946 do 1959)

Računalniki 1. generacije so ENIAC, EDVAC, EDSAC, IAS, UNIVAC I, 701 EDPM in HEATKIT ED-1.

Slabosti 1. generacije so nezanesljivost, majhna zmogljivost, ogromna poraba energije, ogromne fizične dimenzije in visoka cena.



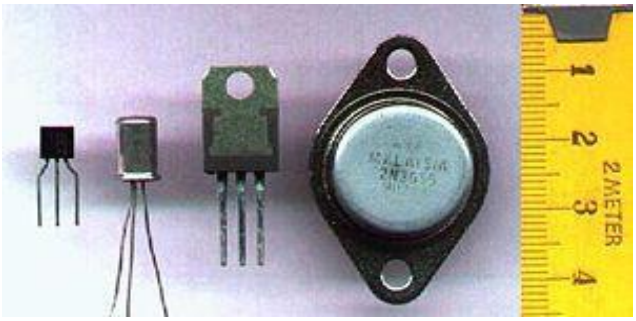
Slika 18: ENIAC
Vir: https://en.wikipedia.org/wiki/ENIAC (14. 8. 2015)

### 3.3.2 2. generacija (od 1959 do 1965)

Elektronski računalniki druge generacije vsebujejo tranzistorje, ki so bili osnova za današnjo digitalno tehniko. **Tranzistor** je polprevodniški elektronski element s tremi priključki, ki ga uporabljamo za ojačevanje, preklapljanje in uravnavanje napetosti. Zaradi svoje velikosti, zanesljivosti in ekonomičnosti je nadomestil elektronke.

Pojavijo se visokonivojski programski jeziki, kot so algol, fortran in cobol. Sistemska oprema teh računalnikov je omogočala čedalje enostavnejše delo z vhodno-izhodnimi enotami.

Prednosti 2. generacije so povečana zmogljivost in zanesljivost ter zmanjšana poraba energije. Slabost še vedno ostaja visoka cena.




Slika 19: Tranzistor
Vir: https://sl.wikipedia.org/wiki/Tranzistor (14. 8. 2015)

### 3.3.3 3. generacija (od 1965 do 1975)

V 3. generaciji se pojavijo elektronski računalniki z integriranim vezjem. **Integrirano vezje oz. čip** je mikrovezje, sestavljeno iz množice elektronskih elementov, ki so na skupnem substratu iz polprevodniškega materiala med seboj povezani v električno vezje.

Prednost 3. generacije je, da so hitrejši, bolj zmogljivi in zanesljivi, hkrati pa manjši, porabijo manj energije, so cenejši in zaradi novih konceptov obdelave podatkov ter operacijskega sistema lahko izvajajo več programov hkrati.

Slika 20: Integrirano vezje

### 3.3.4 4. generacija (od 1975 do danes)

Leta 1971 je Ted Hoff izumil mikroprocesor Intel 4004 in pojavili so se **osebni računalniki**. Prednost je, da so zelo zmogljivi in zanesljivi, majhni ter da porabijo melo energije.

Slika 21: Osebni računalnik

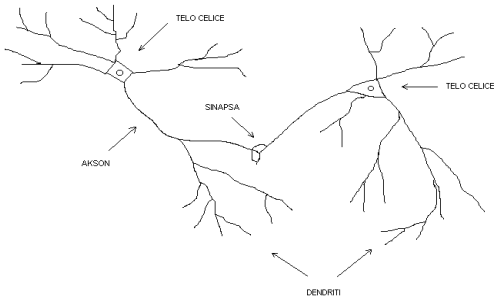
### 3.3.5 5. generacija (od 1981 do danes)

Računalniki znajo komunicirati z uporabnikom v naravnem govornem jeziku in omogočajo shranjevanje ogromnih količin podatkov. Računalnik zna inteligentno sklepati in obdelovati slike ter videti predmete, kot jih vidimo ljudje.

V računalnikih 5. generacije je vgrajenih več procesorjev, obdelava podatkov poteka paralelno, prisotne so optične tehnologije in programiranje z učenjem.

### 3.3.6 6. generacija (od 1984 do neskončnosti)

Računalniki uporabljajo **nevronske mreže**, ki posnemajo človeške možgane.




Slika 22: Nevronske mreže
Vir: http://www.ro.feri.uni-mb.si/predmeti/krmilna/predavanja/krtn.html (14. 8. 2015)

## 3.4 Povzetek

V tem poglavju smo se seznanili z zgodovino računalništva od štetja s pomočjo prstov, kamenčkov, palic itd. do uporabe nevronskih mrež. Podrobneje smo spoznali delovanje abaka, prvih računskih tablic, logaritemskega računala, pascaline, Leibnizevega računala, diferenčnega in analitičnega stroja, elektromehanskih strojev ter generacije elektronskih računalnikov.

## 3.5 Vprašanja in naloge za preverjanje znanja

1. Blaise Pascal je izumil:
   a) abak,
   b) pascline,
   c) diferenčni stroj.
2. Luknjana kartica je:
   a) elektromehanski stroj,
   b) računalnik za sortiranje in tabeliranje podatkov,
   c) vrsta pomnilnika za shranjevanje podatkov.
3. Računalnik je bil v začetku zasnovan kot naprava za:
   a) Računanje,
   b) zbiranje in urejanje podatkov,

c) merjenje.

4. Mehanski računalnik je:

   a) računalnik, ki za pogon uporablja elektromotorje,

   b) računalnik, ki deluje kot mehanizem po principih mehanike.

5. Binarni električni mehanski računalnik Z1 je izdelal:

   a) Blaise Pascal,

   b) Charles Babbage,

   c) Kondar Zuse.

6. Mehaničnega računalnika ni izdelal:

   a) Blaise Pascal,

   b) John von Neumann,

   c) Charles Babbage,

   d) Kondar Zuse.

7. Značilnost prve generacije elektronskih računalnikov je:

   a) nizka cena,

   b) majhne fizične dimenzije,

   c) majhna poraba energije,

   d) majhni zmogljivost in zanesljivost.

8. V obdobju elektronskih računalnikov poznamo:

   a) eno generacijo,

   b) štiri generacije,

   c) šest generacij,

   d) osem generacij.

9. Tranzistor se je pojavil v:

   a) 2. generaciji,

   b) 4. generaciji,

   c) 5. generaciji.

10. Idejni oče računalniškega modela, ki je sestavljen iz centralne procesne enote, pomnilne enote in vhodno/izhodne enote, je:

    a) Kondrad Zuse,

    b) John von Neumann.

# 4 OPERACIJSKI SISTEMI

**Operacijski sistem** (OS) je najpomembnejša programska oprema na računalniku. Upravlja pomnilnik, procese na računalniku ter vso programsko in strojno opremo. Prav tako omogoča komunikacijo med uporabnikom in računalnikom, in sicer ne da bi uporabnik poznal strojni jezik (jezik računalnika).

## 4.1 Naloga operacijskega sistema

Operacijski sistem računalnika upravlja vso programsko in strojno opremo na računalniku. Večino časa se na računalniku sočasno izvaja veliko različnih računalniških programov, ki potrebujejo dostop do centralne procesne enote in pomnilnika. Operacijski sistem koordinira delovanje računalnika in poskrbi, da vsak program prejme, kar potrebuje.

## 4.2 Vrste operacijskih sistemov

Operacijski sistem je po navadi nameščen na računalniku, že preden ga kupimo. Večina ljudi uporablja operacijski sistem, ki je že nameščen na računalniku, vendar pa je mogoče operacijski sistem tudi nadgraditi ali naložiti katerikoli drug operacijski sistem, ki je predstavljen v nadaljevanju. Trije najpogostejši operacijski sistemi za osebne računalnike so:

- **Microsoft Windows**,
- **Apple Mac OS X** in
- **Linux**.

Slika 23: Logotipi najpogosteje uporabljanih operacijskih sistemov

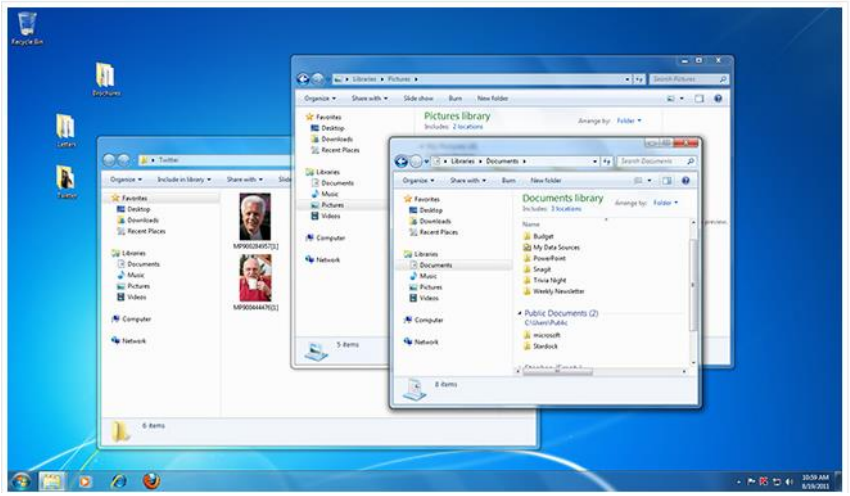
Sodobni operacijski sistemi uporabljajo grafični uporabniški vmesnik. Ta uporabniku omogoča, da zgolj z uporabo miške klikne na ikono, gumb ali meni. Pred izumom grafičnih

uporabniških vmesnikov je moral uporabnik posamezni ukaz natipkati s pomočjo konzolne aplikacije. Vsak grafični uporabniški vmesnik operacijskega sistema ima drugačen videz, vendar pa so sodobni operacijski sistemi zasnovani za enostavno uporabo, zato je večina osnovnih načel enakih.

Slika 24: Grafični uporabniški vmesnik operacijskega sistema Windows in OS X

## 4.3 Microsoft Windows

Microsoft je ustvaril **operacijski sistem Windows** sredi 80. let. V preteklosti je bilo veliko različic tega operacijskega sistema, zadnji pa so **Windows 10** (2015), **Windows 8** (2012), **Windows 7** (2009) in **Windows Vista** (2007). Windows je prednaložen na večini novih računalnikov, kar je povzročilo, da je postal najpriljubljenejši med operacijskimi sistemi na svetu. Izbiramo lahko med več različnimi izdajami operacijskega sistema Windows, kot so **Home Premium**, **Professional** in **Ultimate**.



Slika 25: Windows 7

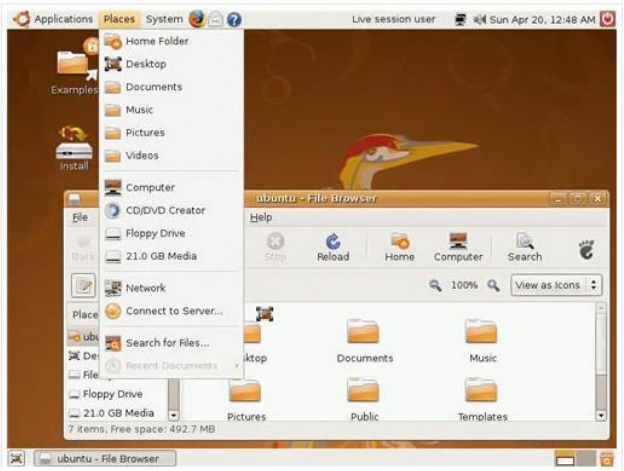
## 4.4 Mac OS X

**Mac OS** je linija operacijskih sistemov, ki jih je ustvarilo podjetje Apple. Operacijski sistem Mac OS je prednaložen na vseh novih računalnikih Macintosh in Mac. Novejše različice so znane kot **OS X**, posebne različice pa vključujejo **Yosemite** (2014), **Mavericks** (2013), **Mountain Lion** (2012), **Lion** (2011) in **Snow Leopard** (2009). Apple ponuja tudi različico **Mac OS X Server**, ki je zasnovana tako, da deluje na strežnikih.

Slika 26: Mac OS X Lion

## 4.5 Linux

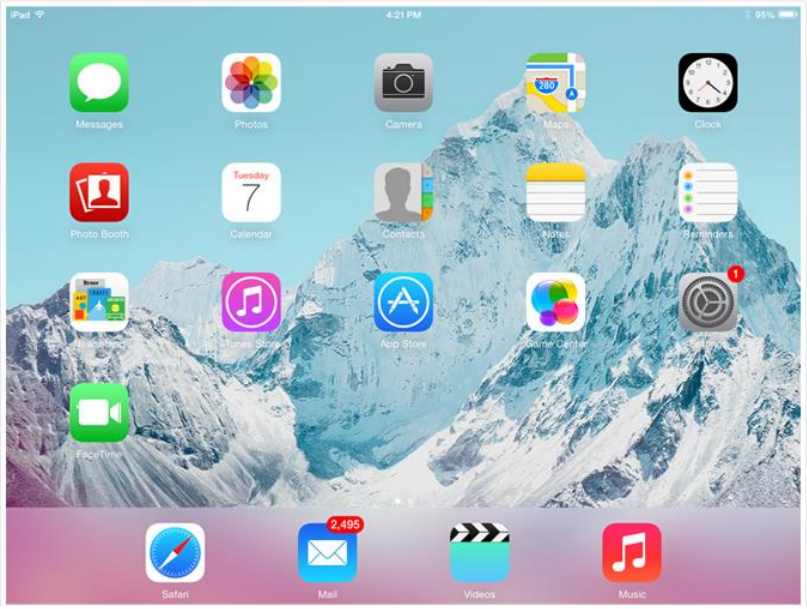
**Linux** je družina odprtokodnih operacijskih sistemov, kar pomeni, da jih lahko kdorkoli spreminja in distribuira. Prednost Linuxa je, da je brezplačen, zato obstaja veliko njegovih različic. Vsaka distribucija ima drugačen videz, najpogostejši pa so **Ubuntu**, **Mint** in **Fedora**.




Slika 27: Linux Ubuntu

## 4.6 Operacijski sistemi za mobilne naprave

Operacijski sistemi, ki smo jih spoznali so namenjeni za namizne ali prenosne računalnike. Mobilne naprave – telefoni, tabličnimi računalniki in MP3-predvajalniki – so drugačne od namiznih in prenosnih računalnikov, zato potrebujejo operacijske sisteme, izdelane posebej za mobilne naprave. Mobilni operacijski sistemi so **Apple iOS**, **Windows Phone** in **Google Android**.




Slika 28: Operacijski sistem Apple iOS

Operacijski sistemi za mobilne naprave ne podpirajo popolnoma vse programske opreme, ki jo podpirajo operacijski sistemi za namizne in prenosne računalnike. Omogočajo nam gledanje filmov, brskanje po spletu, upravljanje koledarja, igranje iger itd.

## 4.7 Datoteke in mape

**Datoteka** je zbirka digitalnih podatkov, shranjenih v spominu kot logična enota. Datoteke se razvrščajo v **mape**. Velikosti datotek in map merimo v kibibajtih (KiB), mebibajtih (MiB) ali gibibajtih (GiB).

### 4.7.1 Datoteke

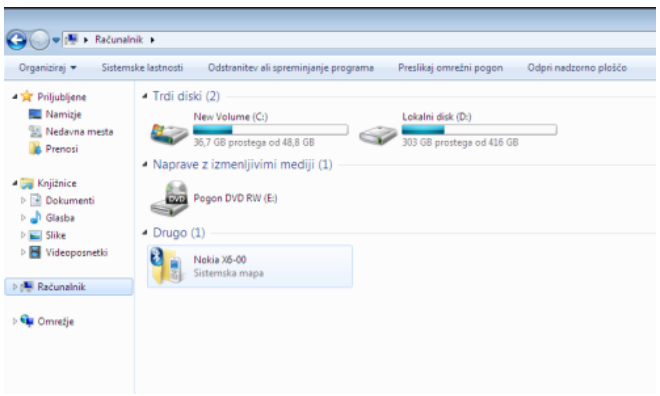
Vsaka datoteka je določena z **imenom** in **končnico**. Končnica označuje vrsto datoteke oziroma nam pove, s katerim programom je izdelana.

Najpogostejše datotečne končnice so:

- .exe – izvršilne datoteke,
- .sys, .dll – sistemske datoteke,
- .tmp, .temp – začasne datoteke,
- .rar, .zip, .7z – stisnjene datoteke,
- .jpg, .jpeg, .gif, .png – slikovne datoteke,
- .mp3, .wav, .wma – avdio datoteke,
- .avi, .wmv, .mpg, .mp4 – video datoteke,
- .txt – besedilne datoteke,
- .rtf, .doc, .docx – datoteke, ustvarjene v programu Microsoft Word,
- .xls, .xlsx – datoteke, ustvarjene v programu Microsoft Excel,
- .ppt, .pptx – datoteke, ustvarjene v programu Microsoft PowerPoint,
- .mdb, .accdb – datoteke, ustvarjene v programu Microsoft Access,
- .pdf – prenosne datoteke in
- .htm, .html, .asp, .aspx, .php – datoteke za splet.

### 4.7.2 Raziskovalec

**Raziskovalec** omogoča delo z datotekami in mapami na računalniku. Datoteke lahko v *Raziskovalcu* poiščemo tako, da uporabimo iskalno polje v naslovni vrstici v zgornjem desnem kotu. Mapa *Moj računalnik* je bila v *Raziskovalcu* v sistemih Windows 8.1 in Windows RT 8.1 preimenovana v mapo *Ta računalnik*.



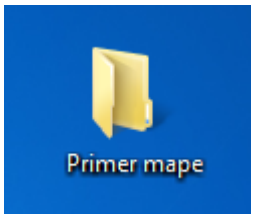

Slika 29: Raziskovalec

Če kliknemo v iskalno polje, se prikaže zavihek *Orodja za iskanje*. Na tem zavihku lahko na primer:

- Poiščemo datoteko na podlagi lastnosti, kot je datum zadnje spremembe, velikost datoteke ali vrsta datoteke. Ko vnesemo iskani izraz, kliknemo zavihek *Orodja za iskanje*, uporabimo možnosti v skupini *Natančneje določi* in nato vnesemo iskane izraze v iskalno polje. Če želimo na primer poiskati datoteke, ki so bile spremenjene v zadnjem tednu, kliknemo *Datum* spremembe in nato *Ta teden*. Rezultate iskanja lahko opredelimo natančneje še z dodatnimi lastnostmi. Če želimo na primer poiskati le imena datotek in ne njihove vsebine, kliknemo *Druge lastnosti* in izberemo *Ime* ter nato vnesemo iskani izraz.
- Če elementov, ki jih iščemo, ne najdemo v določeni knjižnici ali mapi, razširimo iskanje in vključimo različna mesta. Ko se prikažejo rezultati iskanja, kliknemo zavihek *Orodja za iskanje* in v razdelku *Išči znova* izberemo poljubno možnost.
- Če želimo v iskanje vključiti vsebino datotek, kliknemo zavihek *Orodja za iskanje*, *Dodatne možnosti* in nato izberemo *Vsebina datotek*. Če želimo v kazalo dodati mesto, tako da bo v iskanje vključena tudi vsebina datotek, kliknemo zavihek *Orodja za iskanje*, *Dodatne možnosti* in izberemo *Spremeni indeksirana mesta*.

### 4.7.3 Ustvarjanje map in podmap

Za ustvarjenje mape s pritiskom na desni miškin gumb prikličemo meni in iz njega izberemo možnost *Novo* in *Mapa* ter vpišemo ime mape in pritisnemo tipko *Enter*.




Slika 30: Primer mape

Ko je mapa ustvarjena, lahko v njej po zgoraj opisanem postopku ustvarimo novo mapo, imenovano podmapa.

### 4.7.4 Delo z datotekami in mapami

#### 4.7.4.1 Označevanje datotek in map

Datoteke in mape označimo tako, da nanje kliknemo. Označevanje več datotek oziroma map izvedemo na enega od naslednjih načinov:

- pritisnemo in zadržimo levi gumb miške ter vlečemo kazalec miške, dokler ne označimo želene skupine datotek oziroma map,
- držimo tipko *Shift* (potem ko smo datoteko označili) in pritisnemo zadnjo datoteko v zaporedju ter sprostimo tipko *Shift*,
- držimo tipko *Ctrl*, dokler ne označimo vseh želenih datotek oziroma map (ni nujno, da so v zaporedju) in
- vse ikone (datoteke, mape, programe in bližnjice) v določeni mapi označimo s pomočjo menijske vrstice, in sicer *Uredi* in *Izberi vse* ali s kombinacijo tipk *Ctrl + A*.

#### 4.7.4.2 Kopiranje datotek in map

Datoteko ali mapo kopiramo tako, da z desnim miškinim gumbom kliknemo na želeno datoteko oziroma mapo ter izberemo možnost *Kopiraj*, nato pa na želeni lokaciji ponovno kliknemo na desni miškin gumb ter izberemo možnost *Prilepi*. Kopiramo lahko tudi s pomočjo kombinacije tipk **Ctrl + C** ter prilepimo s pomočjo kombinacije tipk **Ctrl + V**.

#### 4.7.4.3 Premikanje datotek in map

Ko želimo premikati datoteko ali mapo, jo najprej poiščemo in označimo ter z desnim klikom iz menija izberemo ukaz *Izreži*, nato kurzor postavimo na mesto, kamor želimo premakniti datoteko ali mapo, ter pritisnemo desni gumb miške in iz spustnega menija izberemo ukaz *Prilepi*. Premikanje z metodo »**povleci-spusti**« izvedemo tako, da označimo datoteko ali mapo, ki jo želimo premakniti, ter na označeni datoteki ali mapi pritisnemo in zadržimo levi gumb miške ter povlečemo kazalec miške na mesto, kamor želimo premakniti datoteko oziroma mapo.

#### 4.7.4.4 Brisanje datotek in map

Datoteko ali mapo lahko izbrišemo na več načinov:

- nanjo kliknemo z desnim miškinim gumbom ter izberemo ukaz *Izbriši*,

- datoteko ali mapo označimo, pritisnemo in zadržimo levi gumb miške ter povlečemo kazalec miške do mape, poimenovane *Koš*,
- označimo datoteko ali mapo in na tipkovnici pritisnemo na **tipko Delete**.

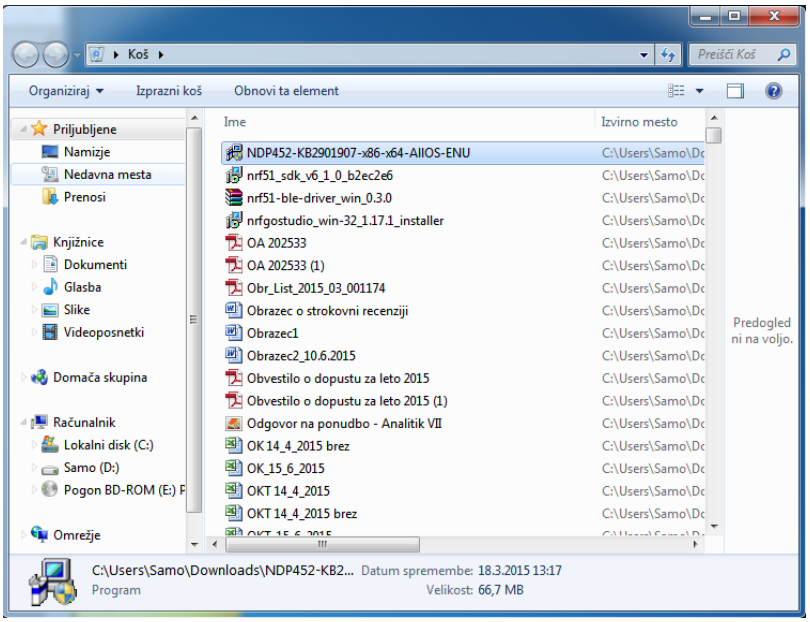
Tako izbrisana datoteka gre v resnici v mapo *Koš*, ki jo nato lahko izpraznimo. Če hočemo datoteko nepovratno izbrisati takoj, pri gornjih operacijah držimo tipko dvigalko (*Shift*).

#### 4.7.4.5 Obnovitev izbrisanih datotek in map iz koša

Če datoteko ali mapo izbrišemo, se še vedno nahaja v mapi *Koš*, od tam pa jo je še vedno mogoče obnoviti ter jo s tem vrniti na njeno izvirno mesto. To storimo tako, da z dvoklikom odpremo pogovorno okno *Koš* in izberemo eno od dveh možnosti:

- obnovi vse elemente: ukaz obnovi in vrne vse datoteke oziroma mape iz mape *Koš* na njihovo izvirno mesto,
- obnovi ta element: ukaz obnovi in vrne le tisto datoteko oziroma mapo, ki smo jo predhodno označili.

V mapi *Koš* imamo tudi možnost ukaza Izprazni koš, ki trajno izbriše vsebovane datoteke in mape.




Slika 31: Mapa Koš

## 4.8 Povzetek

V tem poglavju smo spoznali definicijo operacijskega sistema, naloge in vrste operacijskih sistemov ter se seznanili z operacijskimi sistemi Microsoft Windows, Mac OS X, Linux in operacijskimi sistemi za mobilne naprave. Spoznali smo tudi definicijo datotek in map ter najpogostejše datotečne končnice. Seznanili smo se z ustvarjanjem map in podmap ter označevanjem, kopiranjem, premikanjem in brisanjem datotek in map ter obnovitvijo izbrisanih datotek ali map iz mape Koš.

## 4.9 Vprašanja in naloge za preverjanje znanja

1. Kaj je operacijski sistem?
2. Kateri operacijski sistem uporablja vaš računalnik?
3. Primerjajte operacijska sistema Windows in Mac OS X.
4. Kaj je datoteka in kaj mapa?
5. Katere so najpogostejše datotečne končnice?
6. V mapi *Moji dokumenti* kreirajte mapi *Poročila* in *Obvestila*. V mapi *Poročila* kreirajte dve podmapi, in sicer *Dopisi* in *Tabele*, v mapi *Obvestila* pa podmapi *Učenci* in *Dijaki*. Odprite poljubno mapo na svojem računalniku in postavite pogled na *Podrobnosti*. Datoteke in mape razvrstite po velikosti. V mapo *Poročila* skopirajte največjo datoteko iz mape. Nato preverite število datotek v mapi. Odprite urejevalnik besedil in napišite obvestilo Danes grem v šolo. Ta tekst shranite z imenom *Šola* v mapo *Dopisi*. Napišite novo besedilo Izlet v Postojnsko jamo bo v sredo, 12. 8. 2015. Datoteko shranite pod imenom *Izlet* v isti mapi kot prej. Prekopirajte datoteki *Šola* in *Izlet* iz mape *Dopisi* v mapo *Učenci*, ki se nahaja v mapi *Obvestila*. Odprite dokument *Izlet* v mapi *Dopisi* in popravite datum izleta na današnji datum ter shranite pod imenom *Izlet2* v mapo *Dijaki* ter zaprite program. Premaknite datoteko *Izlet2* iz mape *Dijaki* v mapo *Tabele*. Preimenujte mapo *Učenci* v mapo *Študenti*. Premaknite se v mapo *Poročila* in posnemite zaslonsko sliko. Odprite program Microsoft Word in sliko prilepite v prazen dokument. Dokument shranite v mapo *Poročila* z imenom *Posnetek*. Izbrišite mapi *Poročila* in *Obvestila*.
7. Obnovite mapo z imenom *Poročila* iz *Koša* na računalniku in jo nato ponovno izbrišite.
8. Izpraznite *Koš* na svojem računalniku.

9. Na namizju svojega računalnika ustvarite mapo *Živali*. Na internetu poiščite fotografijo leva in jo shranite pod imenom *Slika leva* v mapo *Živali*, nato pa jo prekopirajte v Word na prazen list. Poiščite tri odstavke besedila o kralju živali in jih prekopirajte pod sliko. Shranite dokument pod imenom *Lev* v mapo *Živali*. Zaprite Word. Odprite mapo *Živali* ter poglejte velikost datoteke *Slika leva*. Izračunajte, koliko takih datotek bi lahko prekopirali na svoj USB-ključ. Rezultat zapišite v *Slikarju* in prostoročno narišite sliko leva. Shranite v mapo *Živali* pod imenom *Portret leva*. Zaprite program. Preverite, koliko takih risbic bi lahko kopirali na svoj USB-ključ. V datoteko *Lev* dodajte še 5 fotografij in 2 strani besedila ter shranite. Preverite velikost in ponovno izračunajte, koliko takšnih datotek bi lahko shranili na svoj USB-ključ. Odgovor dodajte v dokument *Portret leva*.

# 5 UVOD V PROGRAMIRANJE

## 5.1 Algoritem

**Algoritem** je načrt (navodilo) za izvedbo zaporedij akcij oziroma postopkov nad podatki, da bi dosegli želen rezultat. Da bi bil algoritem uporaben, mora biti:

- **natančen**,
- **nedvoumen** in
- za vse možne situacije mora enolično opredeliti **končno zaporedje akcij**, s katerimi dosežemo želeni cilj.

Splošne oziroma **obvezne lastnosti vsakega algoritma** so:

- **Popolnost** – da bi bil algoritem popoln, morajo biti vse njegove akcije natančno definirane.
- **Nedvoumnost** – množica ukazov je nedvoumna le, če obstaja le en način interpretacije ukazov.
- **Determinizem** – če sledi ukazom, je gotovo, da bo želeni rezultat vedno dosežen.
- **Končnost** – izvajanje ukazov se mora (po določenem številu korakov) končati. Zahteva o končnosti se nanaša na število korakov in končno število spremenljivk, ki jih pri tem uporabljamo.

**Splošni atributi algoritmov** so:

- **Splošnost** – algoritem naj rešuje razred problemov, ne le enega.
- **Dobra struktura** – dobro grajen algoritem je narejen iz sestavin oziroma blokov, ki jih je enostavno razložiti, so razumljivi ter jih lahko testiramo in spreminjamo. Sestavine oziroma bloki naj bi bili tako povezani, da lahko enostavno zamenjamo en blok z drugim (boljšim, učinkovitejšim).
- **Učinkovitost** – hitrost, velikost in kompaktnost algoritma. Naprej napišemo algoritem z dobro strukturo, ki v vseh primerih dosega želeni cilj. Šele potem ga izboljšujemo v smislu učinkovitosti.
- **Enostavnost** – algoritem naj bo razumljiv in enostaven za uporabo.
- **Robustnost** – algoritem naj bo trdoživ na napačne podatke.
- **Ekonomičnost** – poceni.

Algoritem lahko predstavimo:

- **Ustno** – algoritem je izražen z besedami.
- **Tabelarično** – algoritem je predstavljen z eno ali več tabel.
- **Z diagramom poteka** (angl. flowchart) – algoritem je predstavljen v obliki diagrama z akcijskimi pravokotniki ali drugimi simboli, ki so povezani s puščicami. Puščice nakazujejo zaporedje akcij, ki naj se izvedejo.
- **S psevdokodo** – algoritem je predstavljen kot množica ukazov, zapisanih v mešanici naravnega jezika in matematične notacije. Oblika ukazov v psevdokodi je podobna postopkovnim programskim jezikom.

## 5.2 Vloga in pomen programskih jezikov

Jezike, v katerih se ljudje pogovarjamo in sporazumevamo, imenujemo **naravni jeziki**. Naravnih jezikov ni mogoče prečistiti, tako da bi bili v izražanju nedvoumni, zato so ljudje načrtno ustvarili **umetne jezike** (matematične izraze, kemijske formule itd.). Tistim umetnim jezikom, ki so namenjeni sporazumevanju oz. upravljanju z računalniki, pravimo **programski jeziki**. Na podlagi tega lahko postavimo osnovno definicijo programskih jezikov. **Programski jeziki** so jeziki, s pomočjo katerih človek podaja navodila oziroma algoritme računalniku.

## 5.3 Delitev programskih jezikov

Programske jezike delimo na pet generacij:

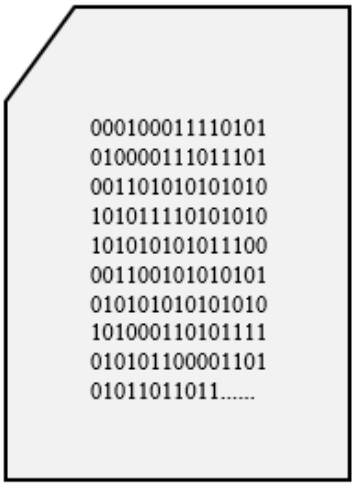
- 1. generacija – strojni jezik,
- 2. generacija – zbirni jezik,
- 3. generacija – višji programski jeziki,
- 4. generacija in
- 5. generacija.

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

### 5.3.1 Strojni jezik – 1. generacija

**Strojni jezik** je najbližje računalniku, saj tako zapisane ukaze računalnik neposredno razume.

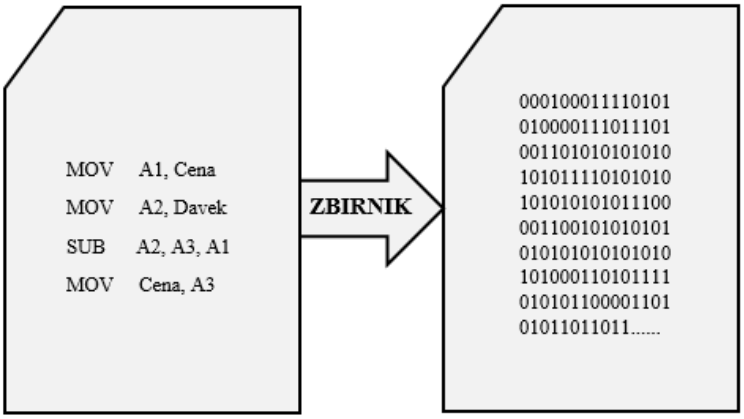
Programiranje s strojnim jezikom je zelo zahtevno, posledica tega pa je velika možnost napak, ki jih je težko odkriti. Vsak procesor ima svoj strojni jezik, saj programi niso prenosljivi med računalniki z različno strojno opremo.




Slika 32: Primer strojnega jezika

### 5.3.2 Zbirni jezik – 2. generacija

**Zbirni jezik** izhaja iz strojnega jezika, vendar pa programa, zapisanega v zbirnem jeziku, procesor neposredno ne razume. Za pretvorbo iz zbirnega v strojni jezik skrbi t. i. **zbirnik**, ki binarne nize, ki predstavljajo ukaze ali elemente procesorja, nadomesti z **mnemoniki**. **Mnemonik** je kratica, ki označuje strojne ukaze. Posamezen ukaz torej nadomestimo z natanko enim mnemonikom.




Slika 33: Primer zbirnega jezika

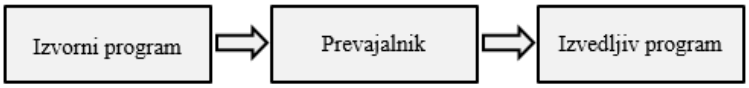
### 5.3.3 Višji programski jeziki – 3. generacija

Ukazi, zapisani v **višjem programskem jeziku**, so podobni stavkom naravnih jezikov, ki so jim dodani matematični ukazi, pri čemer se en ukaz izvede kot zaporedje več stavkov strojnega jezika.

Programiranje z višjim programskim jezikom je učinkovitejše in lažje, programi so razumljivejši, lažje jih je vzdrževati programov in prenesti na drugo strojno opremo. Med višje programske jezike spadajo: **C**, **C++**, **C#**, **java**, **javaScript**, **basic**, **python**, **pascal**, **cobol**, **perl**, **PHP** itd.

V višjem programskem jeziku napisan program moramo pretvoriti v strojni jezik, in sicer lahko to storimo s pomočjo prevajalnika ali tolmača.

**Prevajalnik** je program, ki pretvori izvorni program (običajno zapisan v nekem višjem programskem jeziku) v drug, običajno strojni jezik. Kot rezultat dobimo izvedljivi program. Prevajalnik najprej prevede celoten program in ga nato izvede.




Slika 34: Prevajalnik

**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 kot izvajanje prevedenega programa. Prednost tolmačenega programa pa je

interaktivnost oziroma odzivnost, saj tolmačen program natančno sporoči mesto morebitne napake v programu.

Večina višjih programskih jezikov ima prevajalnik (npr. C++), nekateri imajo tolmača, nekaj višjih programskih jezikov pa ima tudi prevajalnik in tolmača (npr. java, python). Programski jeziki, ki imajo tako prevajalnik kot tudi tolmača, izvorni program najprej prevedejo v vmesno kodo (angl. byte code), nato pa vmesno kodo tolmačijo s posebnim tolmačem t.i. navideznim strojem (angl. virtual machine). Prednost takšnih programskih jezikov je predvsem višja prenosljivost programa oziroma uporabe izdelanega programa na različnih operacijskih sistemih.

### 5.3.4 Programski jeziki 4. generacije

Programski jeziki 4. generacije omogočajo nepomembnost zapisa postopka. Cilj jezikov te generacije je omogočiti programerju komuniciranje s pomočjo abstraktnih pojmov na način, ki je primerljiv z načinom razmišljanja ljudi, ko rešujejo nek problem. Programerju običajno ni treba zapisati algoritma, pač pa se lahko osredotoči na rezultat. Pomembno je, **kaj želi rešiti**, in ne kako. Običajno pa niso splošno namenski, omejeni so na določeno področje:

- podatkovne baze in informacijski sistemi,
- generiranje poročil,
- izdelava spletnih strani,
- izdelava grafičnih uporabniških vmesnikov.

V skupino programskih jezikov 4. generacije spadata **focus** in **SQL**.

Prednosti programskih jezikov te generacije so, da so bliže programerju, programiranje je lažje, manjša je možnost napak in vzdrževanje je enostavnejše. Slabosti teh programskih jezikov pa so, da so omejeni na določeno področje, zahtevajo dodatna znanja in zmogljivejšo opremo ter so manj učinkoviti.

### 5.3.5 Programski jeziki 5. generacije

Mnenja o razvoju programskih jezikov 5. generacije se razlikujejo. Prva možnost razvoja je, da s pomočjo **umetne inteligence** uporabnik vnese razmerje med elementi, program pa sam generira najustreznejšo rešitev. Enostaven primer tega bi lahko bilo ustvarjanje družinskega drevesa, kjer vnesemo osnovne povezave med člani v družini, algoritem pa nato sam določi

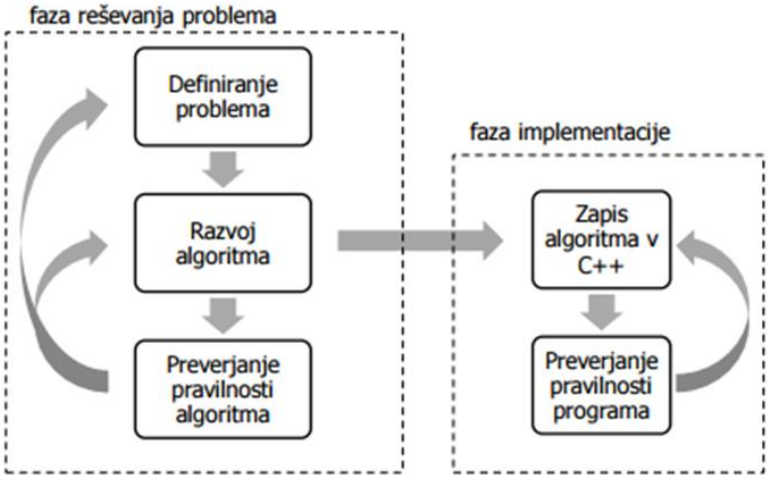
preostale družinske povezave. Druga možnost razvoja, ki je v tem trenutku še nedosegljiva, pa je **uporaba naravnega jezika pri programiranju**.

V skupino programskih jezikov 5. generacije sodita **cash management system** in **prolog**.

## 5.4 Programiranje

**Programiranje** je ustvarjalen proces, saj ne obstaja točno navodilo, kako programirati. Programiranje poteka običajno v dveh fazah:

- **1. faza – 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.




Slika 35: Faze programiranja

## 5.5 Povzetek

V poglavju smo se seznanili s pojmom algoritem ter spoznali splošne lastnosti in splošne atribute algoritmov ter vlogo in pomen programskih jezikov. V nadaljevanju smo spoznali delitev programskih jezikov na generacije. Na koncu poglavja pa smo usvojili še pojem programiranje ter se seznanili s fazami programiranja.

## 5.6 Vprašanja in naloge za preverjanje znanja

1. Kaj je algoritem?
2. Naštej splošne (obvezne) lastnosti algoritmov in jih opiši.
3. Naštej splošne atribute in vsakega opiši.
4. Predstavi vlogo in pomen programskih jezikov.
5. Kako delimo programske jezike?
6. Opiši posamezno generacijo programskih jezikov.
7. Kaj je programiranje?
8. Naštej in opiši fazi programiranja.

# 6 DIAGRAMI POTEKA

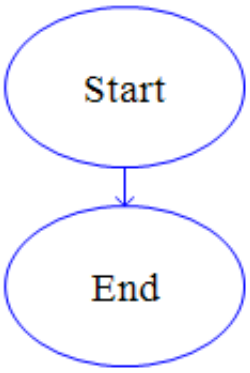
Pri reševanju problemov si največkrat pomagamo z uporabo diagramov poteka. **Diagram poteka** (angl. Flowchart) je vizualna predstavitev toka podatkov. Njegovi standardnimi elementi so:

- začetek ali konec programa,
- računske operacije oz. prireditev ali podprogrami,
- vhodna ali izhodna operacija (branje ali pisanje),
- odločitev ali vejitev in
- tok izvajanja.

Diagrame poteka lahko zelo enostavno in jasno prikažemo s pomočjo programskega okolja **Raptor**, ki je dostopen na spletni povezavi http://raptor.martincarlisle.com/ (14. 8. 2015) in je brezplačen. V nadaljevanju bodo prikazani posamezni elementi in algoritmi izdelani v tem programskem okolju.

## 6.1 Začetek ali konec programa

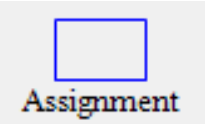
**Začetek ali konec programa** imenujemo tudi terminatora in sta predstavljena z **elipso**.




Slika 36: Začetek in konec programa

## 6.2 Operacije

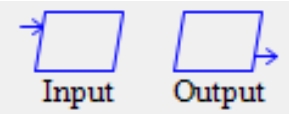
**Računske operacije** oziroma prireditve in podprogrami so predstavljeni s **pravokotnikom**.




Slika 37: Operacije

## 6.3 Vhod in izhod

**Vhodna ali izhodna operacija** (branje ali pisanje) je predstavljena s **paralelogramom**.



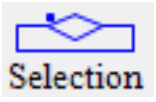

Slika 38: Vhod in izhod

## 6.4 Odločitev ali vejitev

**Odločitev** vsebuje **pogoj** in **dva izhoda**:

- pogoj drži in
- pogoj ne drži.

Odločitve oz. vejitve so predstavljene z **rombom**.




Slika 39: Odločitev ali vejitev

## 6.5 Tok izvajanja

**Tok izvajanja** določa smer premikanja toka, ki je predstavljena s **puščico**. Neformalno velja, da lahko pri risanju izpustimo smer puščice, če je smer premikanja toka jasna, torej če smer premikanja poteka v smeri predhodno nakazane smeri.

Slika 40: Tok izvajanja

## 6.6 Osnovne kombinacije diagramov poteka

Osnovne kombinacije diagramov poteka so:

- zaporedje oz. sekvenca,
- vejitev oz. selekcija in
- ponavljanje oz. iteracija.

Z osnovnimi kombinacijami diagrama poteka lahko sestavimo katerikoli diagram poteka.

### 6.6.1 Zaporedje

**Zaporedje** predstavlja niz standardnih elementov, ki si časovno sledijo eden za drugim tako, da se drugi niz ne more izvesti pred prvim ali za tretjim nizom. Zaporedje si lahko predstavljamo tudi kot nekakšen blok stavkov, ki je predstavljen s pravokotnikom. Za lažje razumevanje uporabe osnovnih kombinacij diagrama poteka si poglejmo naslednji primer zaporedja, kjer želimo sešteti dve vneseni števili ter rezultat preko konzolne aplikacije izpisati. Pred risanjem diagrama poteka rešimo omenjen problem z besedami.

Začetek.

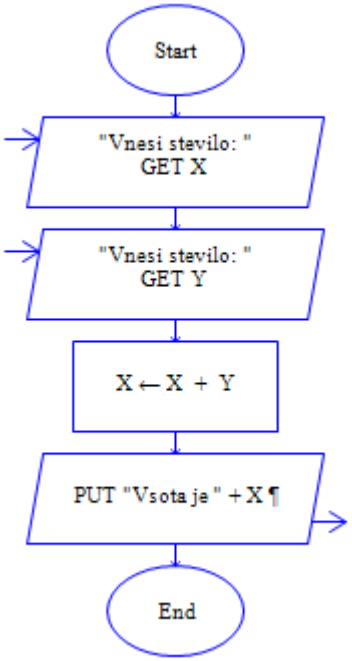
V spremenljivko X zapišemo število, ki smo ga vnesli preko konzolne aplikacije.

V spremenljivko Y zapišemo število, ki smo ga vnesli preko konzolne aplikacije.

Spremenljivki X prištejemo vrednost spremenljivke Y (krajše: X ← X + Y).

Spremenljivko X (zdaj je njena vrednost vsota obeh prebranih števil) izpišemo.

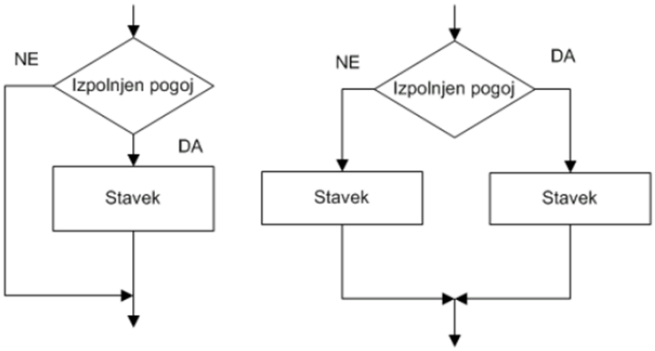
Konec.




Slika 41: Rešitev naloge

### 6.6.2 Vejitev

**Vejitev** omogoča izvajanje izbranih blokov oz. nizov elementov ob določenem pogoju. Vejitev sestavlja element imenovan **pogoj**, ki mora biti zapisan v takšni obliki, da kot rezultat vrne odgovor ***da*** ali ***ne***. Pri rezultatu *da* se vedno izvede določen element ali blok elementov. Vejitev je v literaturi poznana tudi kot **pogojni stavek**.




Slika 42: Dve vrsti vejitev

Za lažje razumevanje si poglejmo primer vejitve, kjer želimo z besedo izpisati, ali je prebrano število negativno, enako 0 ali večje od 0. Pred risanjem diagrama poteka rešimo omenjeni problem z besedami.

Začetek.

V spremenljivko X zapišemo število, ki smo ga vnesli preko konzolne aplikacije.

Preverimo ali je spremenljivka X večja od 0,

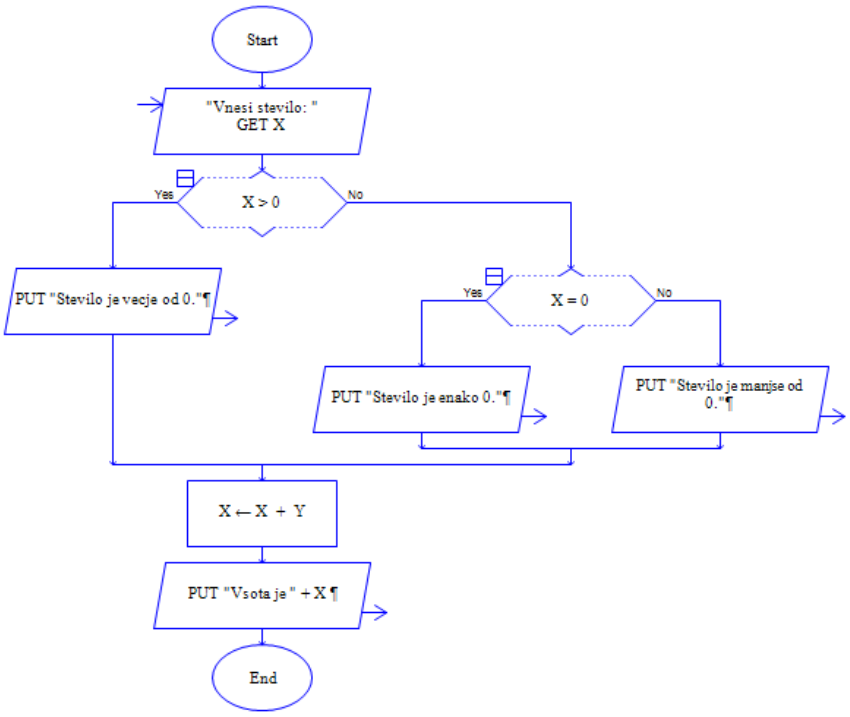
če je, potem izpišemo besedilo “Stevilo je vecje od 0“,

drugače preverimo ali je spremenljivka X enaka 0,

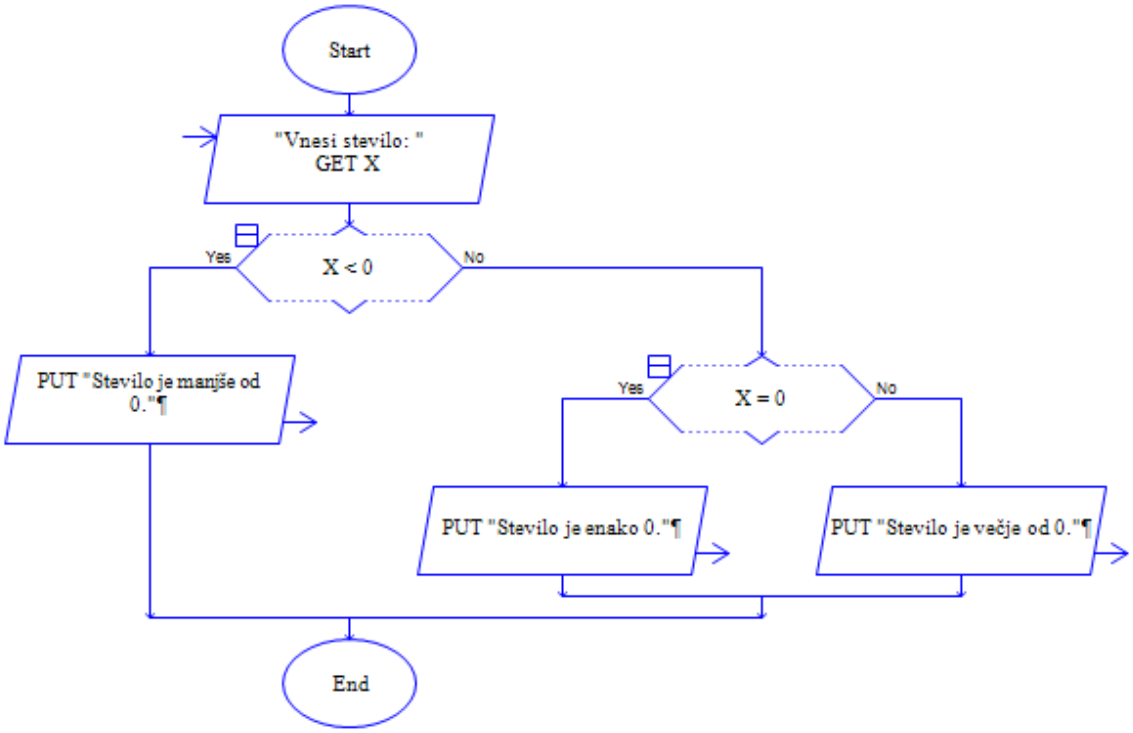
če je, potem izpišemo besedilo “Stevilo je enako 0“,

drugače pa izpišemo besedilo “Število je manjše od 0“.

Konec.



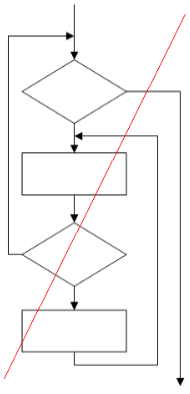

Slika 43: Prva rešitev




Slika 44: Druga rešitev

### 6.6.3 Ponavljanje

**Ponavljanje** se uporablja, kadar želimo iste ukaze večkrat ponoviti. Če se ponavljanje začne znotraj bloka, se mora znotraj bloka tudi končati. Ponavljanja, ki se sekajo, niso dovoljena v strukturiranem programiranju, zato se jih izogibamo tudi pri risanju diagramov poteka.

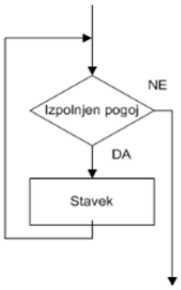


Slika 45: Nedovoljena ponavljanja

Vrste ponavljanj so:

- **stavek while**,
- **stavek do while** in
- **stavek for**.

**Zanka while** se ponavlja, dokler je pogoj izpolnjen, in je najbolj univerzalna zanka, saj sta iz te zanke izpeljani ostali dve zanki, torej zanka do while in zanka for.

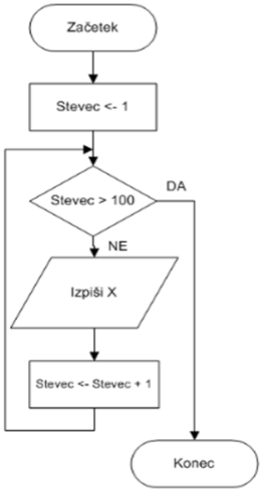



Slika 46: Zanka while

**Zanka do while** se ponavlja, dokler pogoj ni izpolnjen, in jo uporabimo takrat, ko želimo, da se stavek izvede vsaj enkrat.


Slika 47: Zanka do while

**Zanka for** se po navadi uporablja, ko želimo šteti izvajanje operacij. Zato moramo pri zanki for uporabiti **števec**.




Slika 48: Zanka for

Za lažje razumevanje si poglejmo primer uporabe zank, kjer želimo izpisati vsa cela števila od 1 do 100. Pred risanjem diagrama poteka ponovno najprej rešimo omenjeni problem z besedami.

Začetek.

V spremenljivko X zapišimo vrednost 1.

Dokler pogoj X je večji od 100 ni izpolnjen, se izvaja naslednje:

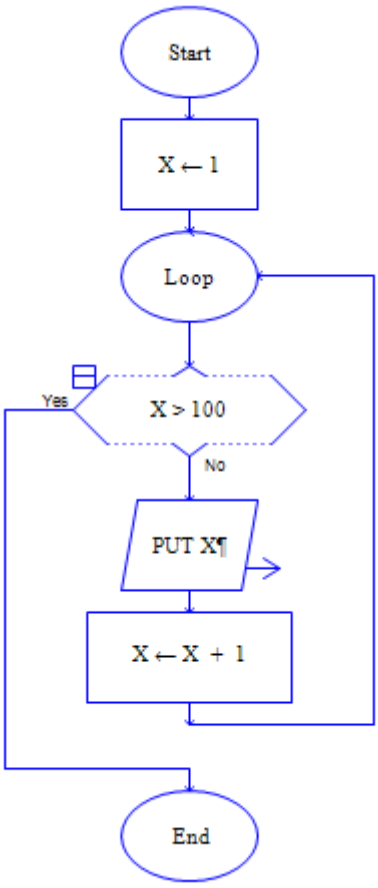
izpišemo spremenljivko X

spremenljivko X povečamo za 1

ponavljamo

Ko je pogoj X večji od 100 izpolnjen, končamo s ponavljanjem izpisa.

Konec.




Slika 49: Rešitev naloge

## 6.7 Povzetek

V tem poglavju smo spoznavali diagrame poteka, ki so ključnega pomena za razumevanje in implementacijo dejanskih problemov. S poznavanjem in razumevanjem diagramov poteka smo na enostaven in razumljiv način spoznali osnovne gradnike programiranja, kar omogoča lažji prehod na programske jezike. Za lažje razumevanje je v poglavju predstavljenih nekaj osnovnih primerov uporabe diagramov poteka v programskem okolju Raptor.

## 6.8 Vprašanja in naloge za preverjanje znanja

1. Narišite diagram poteka, ki sešteje vrednosti *A* in *B* ter izpiše rezultat.
2. Narišite diagram poteka za program, ki od vrednosti *stevilo1* odšteje vrednost *stevilo2* in izpiše rezultat.
3. Narišite diagram poteka za program, ki najprej poveča vrednost spremenljivke *rezultat* za vrednost spremenljivke *vec*, nato pa ga zmanjša za vrednost spremenljivke *manj* ter vrednost spremenljivke *rezultat* izpiše.
4. Narišite diagram poteka za program, ki prebere dve celi števili *x* in *y*, zamenja njuni vrednosti med seboj in ju izpiše.
5. Narišite diagram poteka za program, ki izpiše predhodnika in naslednika vnesenega števila.
6. Narišite diagram poteka za program, ki izpiše vsoto celih števil med 2 in 222.
7. Narišite diagram poteka za program, ki izpiše vsoto sodih števil med 1 in 100.
8. Narišite diagram poteka za program, ki izpiše vsoto lihih števil med 3 in 400.
9. Narišite diagram poteka za program, ki izpiše vsoto celih števil med *A* (prebrano število) in 555.
10. Narišite diagram poteka za program, ki izpiše vsoto celih števil med spremenljivkama *spodnjaMeja* in *zgornjaMeja*, ki sta prebrani števili.

# 7 VIRI IN LITERATURA

A. Krapež, B. Resinovič: Informatika. Delovni zvezek za 1. in 2. letnik srednjih šol. DZS, 2001.

R. Wechtersbach: Informatika. Saji, 2005.

R. Wechtersbach, S. Žust: Računalništvo. Zavod S-Lara, 1999