6.3. Algoritam

Dakle, **algoritam** je slijed radnji ili postupaka koji dovode do rješenja postavljenog zadatka.

Uz svaki algoritam moraju jasno biti definirani **početni objekti** (ulazni podaci) nad kojima se obavljaju operacije (iz primjera: jaja, šećer, brašno, maslac, kuhača, posuda, kuharica). Kao ishod provođenja algoritma pojavljuju se **završni objekti** ili rezultati (iz primjera: kolač).

Algoritam mora biti sastavljen od *konačnog broja koraka* koji ukazuju na slijed operacija koje treba obaviti nad početnim objektima kako bi se dobili rezultati. Svaki korak opisuje se

6.4. Osnovni algoritamski postupci

Postoje tri osnovna algoritamska postupka:

* slijed
* grananje
* ponavljanje (petlja)

6.4.1. Slijed

Naredbe slijede jedna iza druge.

Iz primjera "pečenje kolača":

UZMI zdjelu, kuhaču i vagu

ODIJELI žumanjke od bjelanjaka

DODAJ šećer

DODAJ maslac

6.4.2. Grananje

Kod grananja se prvo postavlja neki uvjet, a zatim se ovisno o rezultatu uvjeta izvode sljedeće naredbe.

Uvjet je uvijek takav da se na njega može odgovoriti sa DA ili NE. Takve uvjete zovemo **logički uvjeti**.

Iz primjera "pečenje kolača":

AKO nemaš sve sastojke,

ONDA odi u trgovinu

INAČE počni peći kolač

6.4.3. Ponavljanje

Kod ponavljanja određeni niz naredbi izvršava se zadani broj puta.

Pri tome možemo imati dvije situacije:

* prije početka ponavljanja unaprijed se točno zna koliko puta postupak treba ponoviti,
* broj ponavljanja ovisi o izvršavanju niza naredbi koje se ponavljaju i tada se broj ponavljanja ne zna unaprijed.

Niz naredbi koje se ponavljaju zajedno s naredbama koje određuju ponavljanje najčešće zovemo **petlja**.

Iz primjera "pečenje kolača":

DOK tijesto nije pečeno

peci na laganoj vatri

DODAJ po jednu žlicu

ili

snijega

brašna

DOK ima sastojaka

Program se sastoji od niza naredbi, a algoritam od niza instrukcija.

Instrukcije se mogu podijeliti u nekoliko skupina.

6.5. Dijagram tijeka

Algoritme za rješavanje problema najčešće prikazujemo grafički pomoću dijagrama tijeka.

**Dijagram tijeka** je grafički prikaz algoritma koji olakšava izradu programa.

On nam omogućuje da što jednostavnije i točnije napišemo program i da se u njemu ne pojavljuju greške.

Tako prikazan algoritam vrlo je pregledan i potpuno određen. Posebno je pogodan za analize programa, traženje sličnih rješenja ili traženje potrebnih izmjena.

Pri crtanju dijagrama tijeka služimo se simbolima:

|  |  |
| --- | --- |
|  | početak, kraj i prekid programa |
|  | ulaz podataka |
|  | izlaz podataka |
|  | naredba (operacija) |
|  | grananje (odluka) – uvijek ima jedan "ulaz", a dva "izlaza" |
|  | linije toka programa |

6.6. Primjeri i zadaci

Primjer 1: Napiši algoritam i sastavi dijagram tijeka za program koji će učitati dva broja pa ispisati koji je veći.

|  |  |
| --- | --- |
| ALGORITAM | DIJAGRAM TIJEKA |
| početak;  upiši A,B;  ako je A>B onda ispiši A  inače ispiši B;  kraj | NE  A, B  B  A > B  A  DA |

6.6.3. Ponavljanje (petlja)

Primjer 4: Napiši algoritam i sastavi dijagram tijeka za program koji će učitati jedno ime, a zatim 25 puta ispisati da je ″ime″ pametan.

|  |  |
| --- | --- |
| ALGORITAM | DIJAGRAM TIJEKA |
| početak;  upiši IME;  ponoviti 25 puta  ispiši IME, ″ je pametan″;  kraj | IME  IME, ″je pametan″  ponavljaj  25 puta |
| početak;  upiši IME;  BR = 1;  dok je BR<=25 radi  ispiši IME, ″ je pametan″;  BR = BR + 1;  kraj | IME  IME, ″je pametan″  BR = 0  BR<=25  DA  NE |

Faze programiranja

Postupak izrade programa nije nimalo lak posao. Često je potrebno mnogo znanja, snalažljivosti i dobrih ideja da bismo mogli izraditi dobar program. Zato postupak izrade programa dijelimo u nekoliko faza:

**1. RAZUMJEVANJE PROBLEMA**

Da bi uopće mogli krenuti, potrebno je dobro razumjeti zadatak. Moramo znati što se traži u zadatku i predvidjeti sve slučajeve koji se mogu dogoditi pri obradi.

**2. OPIS ULAZNIH I IZLAZNIH PODATAKA**

Moramo točno znati koji podaci su nam zadani (ulazni), kojeg su tipa, postoje li neka ograničenja za ulazne podatke. Također je potrebno znati i što je nepoznato, odnosno koje podatke ćemo odrediti programom (izlazni podaci) i kojeg su tipa.

Ovom se fazom utvrđuje ŠTO se treba raditi a ne kako to učiniti.

**3. ODREDITI ALGORITAM**

Kada točno znamo ŠTO treba raditi, potrebno je odrediti metodu, način rješavanja, odnosno KAKO riješiti zadatak. Tada tražimo dobar, brz i efikasan postupak za rješavanje problema.

Pritom treba paziti na redoslijed izvođenja radnji:

* ulaz podataka
* obrada podataka
* izlaz podataka

**4. NACRTATI DIJAGRAM TIJEKA**

Algoritam kojim ćemo riješiti problem prikazujemo grafički.

**5. PISANJE PROGRAMA U NEKOM PROGRAMSKOM JEZIKU (KODIRANJE)**

Opisani tijek programa sada prevodimo u naredbe nekog višeg programskog jezika, odnosno pišemo program na računalu.

**6. PREVOĐENJE PROGRAMA**

Program treba prevesti iz višeg programskog jezika u strojni jezik kojeg računalo razumije. To rade programi prevoditelji (kompajleri).

**7. IZVRŠAVANJE I TESTIRANJE PROGRAMA**

Zatim po prvi puta možemo pokrenuti program na računalu. Moramo ga isprobati za različite ulazne podatke, birajući ih tako da obuhvatimo najrazličitije mogućnosti ulaznih podataka. Ako naiđemo na neku grešku, tada je ovo faza u kojoj sve greške u programu treba ispraviti.

**8. PISANJE PROGRAMSKE DOKUMENTACIJE**

Nakon što napišemo nekoliko programa, a posebno ako su poduži, nakon nekog vremena nećemo se sjećati koji smo točno postupak koristili za rješavanje kojeg od tih problema. Zato je važno tijekom izrade programa pisati programsku dokumentaciju u kojoj su opisani svi koraci izrade programa.

**9. ODRŽAVANJE PROGRAMA**

Napisani program po potrebi se mora dotjerivati i prilagođavati novim zahtjevima korisnika i novoj opremi.

### POJAM ALGORITAM

**ALGORITAM – postupak ili pravilo za rješavanje određene vrste problema**

– svodi se na izvođenje pojedinih osnovnih operacija **zadanim redoslijedom**

– sastoji se od opisa skupa koraka

**PRIMJERI algoritama:**

* 1. **prelaženje ulice**: redoslijed
     1. 1. pogledaj lijevo
     2. 2. pogledaj desno
     3. 3. ako ide vozilo stani
     4. 4. ako nema vozila - prijeđi
  2. **kuharski recepti**
  3. **zbrajanje decimalnih brojeva**

### Osnovni algoritamski konstrukti:

**1. SLIJED (SEKVENCA)**: skup jednog ili više koraka algoritma koji se **jedan za drugim** broj koraka je proizvoljan

PRIMJER: 1) unesi n vrijednosti i zbroji ih

2) izračunaj prosječnu vrijednost

3) ispiši zbroj i prosječnu vrijednost

**2. GRANANJE (SELEKCIJA): odluka koju od dvije ili više sekvenci treba izvršiti s obzirom na postavljeni uvjet**

UVJET istina S1

neistina S2

IF uvjet THEN S1

ELSE S2

IF .. THEN.. ELSE konstrukt većine programskih jezika visoke razine

**3. PONAVLJANJE (ITERACIJA): slijed algoritamskih koraka koji se provodi određeni broj puta, sve dok se ne ispuni** (ili dok je određeni uvjet neispunjen)

Petlje FOR, WHILE, REPEAT

### Uvjeti koje mora zadovoljavati algoritam:

1. točnost (daje točan ishod za sve moguće ulaze i problemske situacije)
2. brzina (algoritam mora dati točan ishod u zadovoljavajućem vremenskom periodu)
3. razumljivost (mora biti razumljiv i primjenjiv i u drugim okolnostima

### Način prikaza algoritma:

1. grafički (korištenjem posebnih simbola postaje univerzalan)
2. pisan u obliku koraka
3. pseudokod

### Elementi dijagrama toka:

|  |  |
| --- | --- |
|  | Početak ili kraj algoritma |
|  | Unos podataka, prikaz rezultata |
|  | Jednostavna radnja (naredba) |
|  | Odluka ili grananje |
|  | Tok programa |
|  | Nastavak dijagrama |