

# Ohjelmoinnin perusteet

## R0027



# D

## Algoritmit



**Viopen luku 2**

# Algoritmi

---

- ❑ Ohjelmointiongelmalle kyetään kehittämään looginen ratkaisumalli
- ❑ Täytyy siis selvittää toimintatavat, joilla lopputulokseen päästään
- ❑ Algoritmi on suoritusohje tietokoneelle
- ❑ Algoritmilla päästään haluttuun lopputulokseen

# Algoritmi leivontaan

---

Etsi kulho (mieluiten puhdas)

Kaada kulhoon vettä, jauhoja, kananmuna, suolaa ja hiiva

Vatkaa ainekset sekaisin kauhalla

Lisää jauhoja ja unohda kauha

Jatka vatkaamista käsin, kunnes taikina vaikuttaa sopivalta

Jätä taikina kymmenen uutisten ajaksi rauhaan

Laita uuni päälle

Tee uunipellille 10 yhtä suurta palleroa ja laita pelti uuniin

Paista kunnes sämpylät ovat ruskeita

Maista

# Ongelmia leivonnassa

---

- ❑ Liikaa suolaa
- ❑ Paloivat koppuraisiksi
- ❑ Vaikea irroittaa uuninpellistä ilman leivinpaperia

= >

- ❑ Algoritmi tulisi miettiä hyvin tarkkaan etukäteen
- ❑ Algoritmi on suoritusohje, eikä se synny tekemisen ohella
- ❑ Algoritmi ei voi olla epämääräinen ja tulkinnanvarainen

# Pseudokoodi

---

- ❑ Ohjelmointikielen kaltaista
- ❑ Vapaamuotoista kuvausta
- ❑ Ei tarkkaa kielioppia eli syntaksia
- ❑ Määrittelee ohjelman loogisen etenemisen
- ❑ Sopii minkä tahansa ohjelmointikielen osaajalle

Pseudokoodi  
on yksi tapa  
hahmotella  
algoritmin  
toimintaa

# Pseudokoodi

## Peräkkäisrakenne

---

herää  
syö aamiainen  
mene kouluun  
opiskele  
palaa koulusta  
tee läksyt  
käy iltapesulla  
mene nukkumaan

# Pseudokoodi

## Valintarakenne

---

Jos sinulla on rahaa, käy elokuvissa

```
JOS (sinulla on rahaa)
{
    käy elokuvissa
}
```



# Pseudokoodi

## Valintarakenne

---

- Valintarakenne voidaan liittää myös peräkkäisrakenteen sisälle

```
herää  
syö aamiainen  
mene kouluun  
opiskele  
palaa koulusta  
JOS (sinulla on rahaa)  
{  
    käy elokuvissa  
}  
käy iltapesulla  
mene nukkumaan
```

# Pseudokoodi

## Valintarakenne

---

- Jos-rakenne voi sisältää useita toimintoja

```
JOS (sinulla on rahaa)
{
    käy elokuvissa
    käy nakkikioskilla
}
```

# Pseudokoodi

## Valintarakenne

---

- JOS-rakenne voi sisältää vaihtoehtoisen MUUTEN-haaran

```
JOS (sinulla on rahaa)
{
    käy elokuvissa
    käy nakkikioskillä
}
MUUTEN
{
    katso tv:tä
}
```

# Pseudokoodi

## Valintarakenne

---

- Rakenteet voivat olla myös sisäkkäisiä

```
JOS (sinulla on rahaa)
{
    käy elokuvissa
    käy nakkikioskillä
}
MUUTEN
{
    JOS (tv:stä tulee jotain katsomisen arvoista)
    {
        katso tv:tä
    }
    MUUTEN
    {
        käy lenkillä
    }
}
```

# Pseudokoodi

## Toistorakenne

---

Kaiva kuoppaa, kunnes se on tarpeeksi syvä

NIIN KAUN KUIN (kuoppa on liian pieni)

{

    kaiva lapiollinen

}

# Pseudokoodi

---

## **Esimerkki:**

*Tee algoritmi, joka osaa laskea  
kaksi käyttäjän antamaa lukua yhteen ja  
tulostaa summan näytölle*

## **Toteutus pseudokoodilla:**

```
kysy käyttäjältä numero X  
kysy käyttäjältä numero Y  
laske Z:lle arvo  $X+Y$   
tulosta näytölle Z
```

# Pseudokoodi

---

## Esimerkki

*Tee algoritmi, joka osaa laskea kaksi käyttäjän antamaa lukua yhteen. Älä anna ohjelman hyväksyä muita syötteitä kuin numeroita.*

## Toteutus pseudokoodilla

```
NIIN KAUAN KUIN (X ei ole numero)
    Kysy käyttäjältä numero X

NIIN KAUAN KUIN (Y ei ole numero)
    Kysy käyttäjältä numero Y

laske Z:lle arvo  $X+Y$ 

tulosta näytölle Z
```

# Pseudokoodi

## Valintarakenne

---

- Rakenteet voivat olla myös sisäkkäisiä

```
KYSY LUKU X
KYSY LUKU Y

JOS (X == Y)
{
    TULOSTA "OVAT YHTÄSUURIA"
}
MUUTEN
{
    JOS (X > Y)
    {
        TULOSTA "EKA OLI SUUREMPI"
    }
    MUUTEN
    {
        TULOSTA "TOKA OLI SUUREMPI"
    }
}
}
```



# Vuokaavio

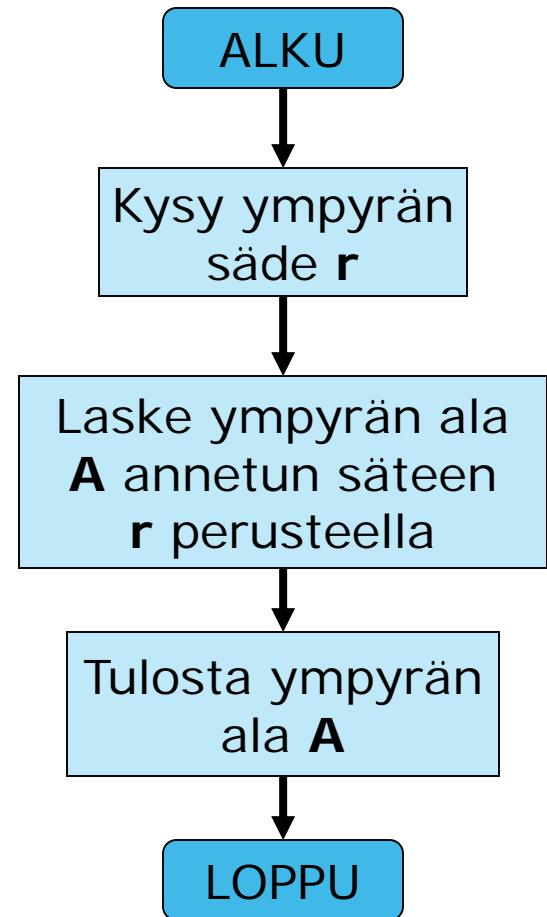
## Ympyrän alan laskeminen - Vaihe 1

- *Laske ympyrän ala, kun säde annetaan mielivaltaisesti ja se on suurempi tai yhtä suuri kuin yksi ja pienempi tai yhtäsuuri kuin 100*

- $A = \pi \cdot r^2$

- $\pi = 3,14159265$

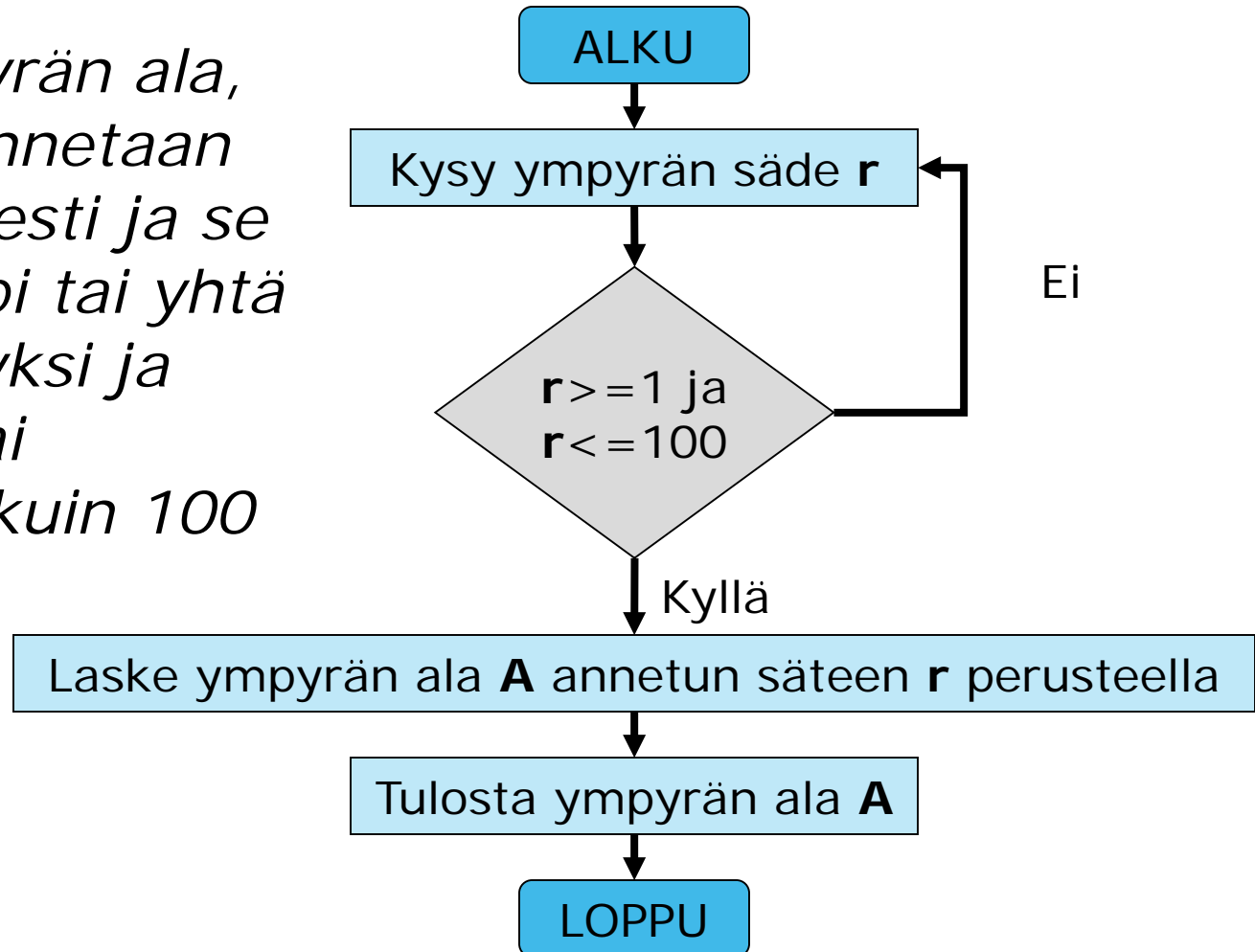
Vuokaavio on yksi tapa hahmotella algoritmin toimintaa



# Vuokaavio

## Ympyrän alan laskeminen - Vaihe 2

- *Laske ympyrän ala, kun säde annetaan mielivaltaisesti ja se on suurempi tai yhtä suuri kuin yksi ja pienempi tai yhtäsuuri kuin 100*

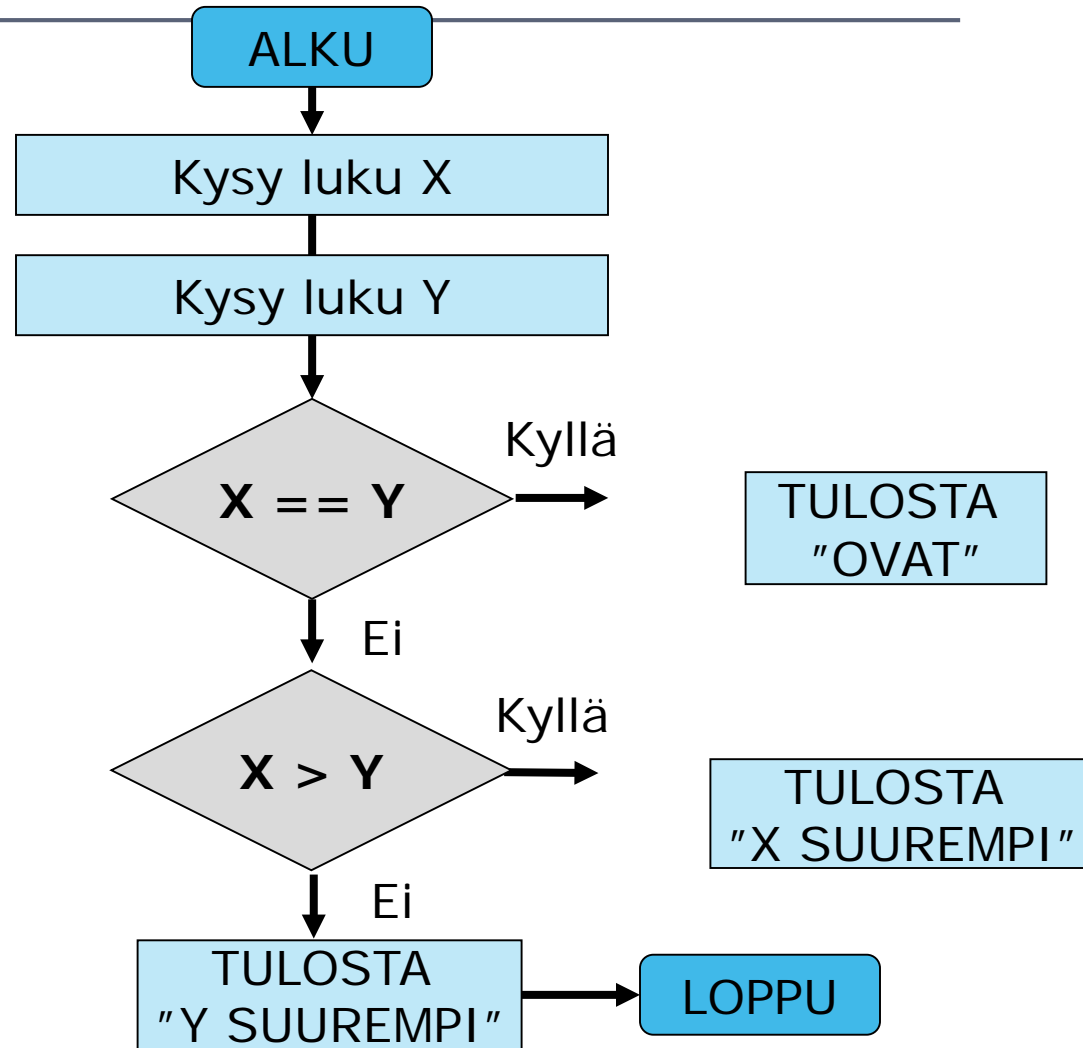


# Vuokaavio

## Algoritmi vuokaaviona

*Hahmottele seuraava algoritmi pseudokoodia käyttäen:*

*Algoritmi kysyy käyttäjältä 2 lukua ja tulostaa ovatko ne yhtäsuuria. Algoritmi tulostaa myös sen, kumpi luvuista on suurempi.*



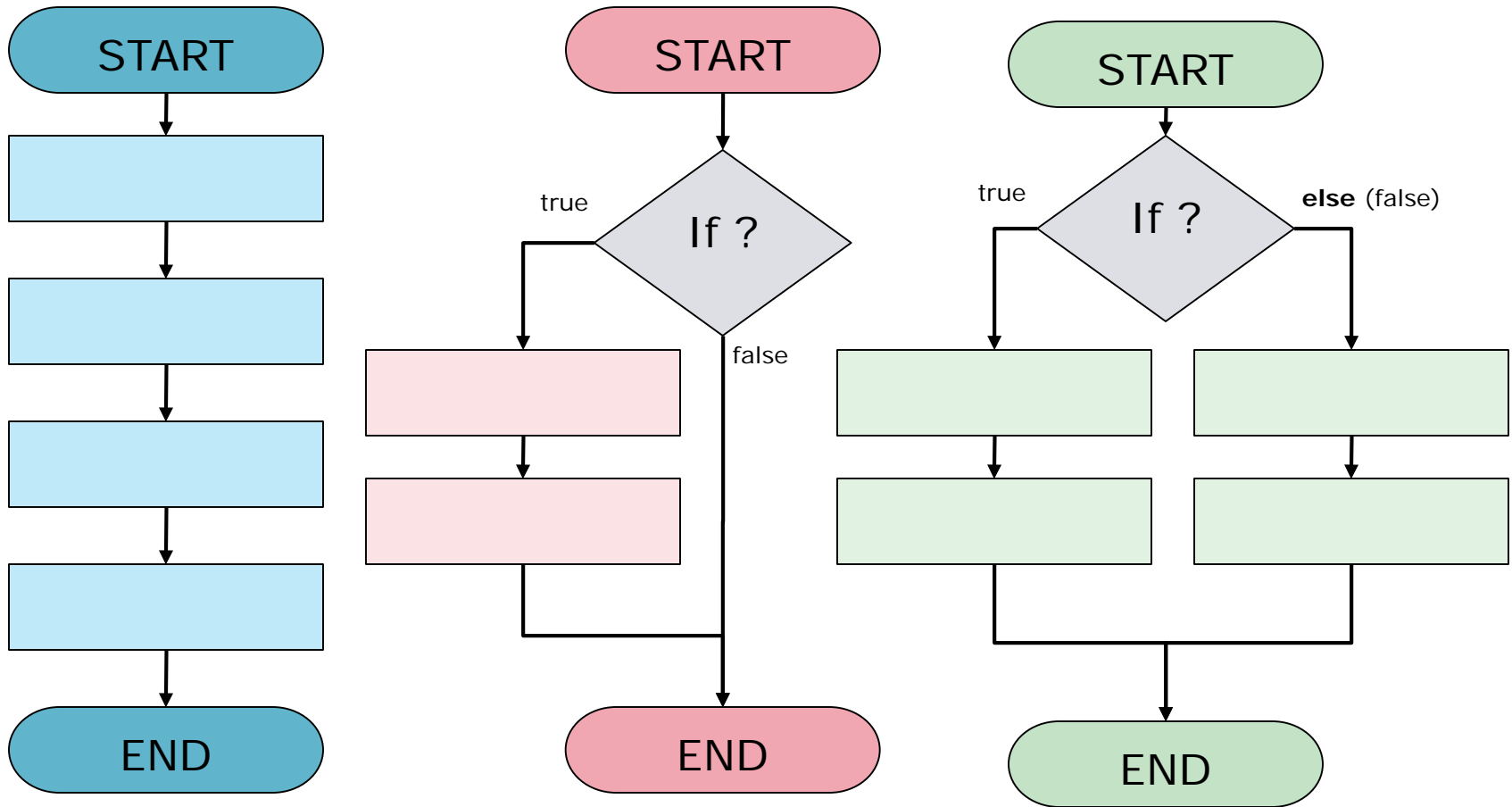
# D

## Valintalauseet ja toistolauseet (do-while)

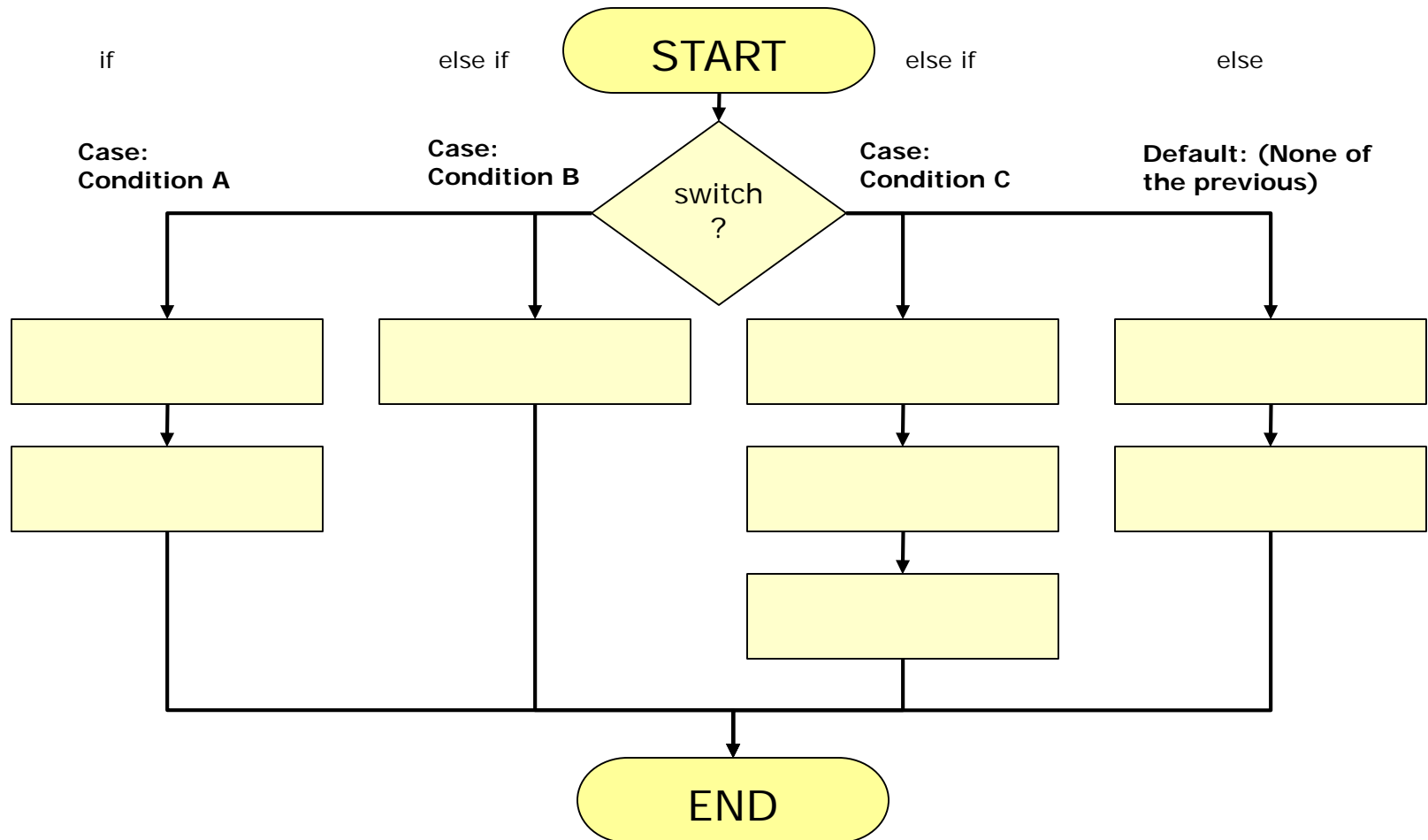


**Viopen luku 6 (alkuosa)**

# Vaihtoehtoiset polut (if-else)



# Vaihtoehtoiset polut (switch)(if-else if)



# Rakenteiset lauseet

---

- ❑ Rakenteisilla lauseilla voidaan vaikuttaa lähdekoodirivien suoritusjärjestykseen
- ❑ Rakenteiset lauseet määrittävät metodin sisälle uusia lohkoja, jotka alkavat ja loppuvat aaltosulkuihin { }
- ❑ **HUOM! Lohkon sisällä määritelty muuttuja ei ole näkyvissä lohkon ulkopuolelle**

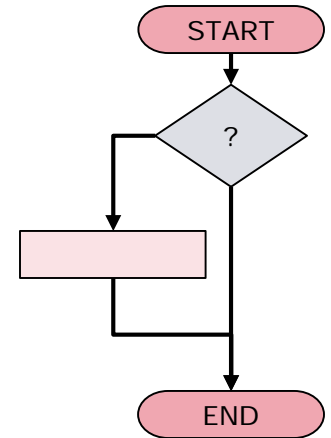
# Valintalause - if

---

- Jos tilillä on rahaa, tilaa pitsa.

```
boolean tilillaOnRahaa = true;  
  
if (tilillaOnRahaa) {  
    System.out.println("Tilaa pitsa.");  
}
```

```
if (luku1 > luku2) {  
    System.out.println("Luku1 on suurempi kuin luku2");  
}
```

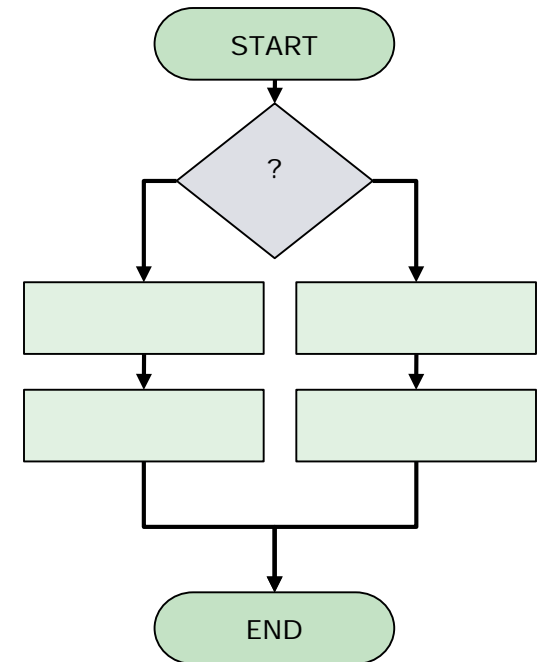




# Valintalaus - else

- Jos tilillä on rahaa, tilaa pitsa.  
Muuten mene lenkille.

```
if (tilinSaldo > 0) {  
    System.out.println("Tilaa pitsa.");  
    System.out.println("Katso leffa.");  
}  
else {  
    System.out.println("Mene lenkille");  
    System.out.println("Käy suihkussa");  
}
```



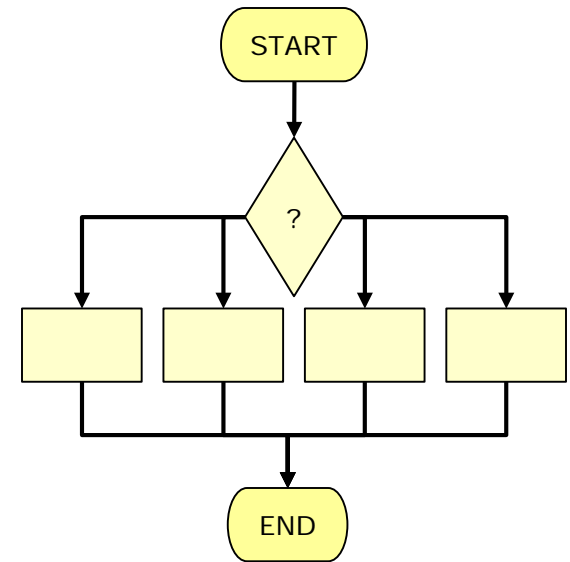
# Valintalause – switch case

---

```
switch (kortinArvo) {  
  case 1:  
    System.out.println("Kortti on Ässä. ");  
    break;  
  case 13:  
    System.out.println("Kortti on Kuningas. ");  
    break;  
  case 12:  
    System.out.println("Kortti on Rouva. ");  
    break;  
  case 11:  
    System.out.println("Kortti on Jätkä. ");  
    break;  
  default:  
    System.out.println("Kortti on väliltä 2-10. ");  
}
```

# Valintalause – else if

```
if (month == 1) {  
    System.out.println("It is cold out there!");  
}  
else if (month == 6 || month == 7) {  
    System.out.println("It is summer!");  
}  
else if (month == 12) {  
    System.out.println("Santa is coming!");  
}  
else {  
    System.out.println("School/work time!");  
}
```



# Merkkijonojen vertailu

---

HUOM. Merkkijonojen yhtäsuuruutta ei voi verrata == operaattorilla!

Tätä varten on käytettävä equals-metodia.

```
String teksti = "kurssi";
```

```
String toinenTeksti = "pursi";
```

```
if ( teksti.equals(toinenTeksti) ) {  
    System.out.println("Samat tekstit!");  
}  
else {  
    System.out.println("Ei samat tekstit!");  
}
```

# Toistolauseet (do-while)



# Pseudokoodi

## Toistorakenne

---

Kaiva kuoppaa, kunnes se on tarpeeksi syvä

```
NIIN KAUN KUN (kuoppa on liian pieni) {  
    kaiva lapiollinen  
}
```

```
TEE UUDELLEEN {  
    kaiva lapiollinen  
} NIIN KAUAN KUIN (kuoppa on liian pieni)
```

# Toistolauseet

---

- ❑ Tietokone soveltuu erinomaisesti toistamaan asioita väsymättömästi kerrasta toiseen.
- ❑ Toistolauseella saamme ohjelman suorituksen toistamaan tiettyä tehtävää niin kauan kun annettu ehto on tosi.
- ❑ Toistolauseen suorituksessa annettu ehto tarkistetaan
  - ennen toistettavien lauseiden suorittamista  
**while, for**
  - tai niiden suorittamisen jälkeen  
**do-while**

# while - syntaksi

---

- Java-kielessä while-lauseen kieliopillinen muoto voidaan tiivistää seuraavaan esitykseen

```
while (ehto) {  
    lauseita; //voi olla yksi tai useampi  
}
```



# while - toiminta

---

## □ Vertaa seuraavia rakenteita

```
if (rahaa > 100) {  
    ostaLevy();  
}
```

```
while (rahaa > 100) {  
    ostaLevy();  
}
```

```
while (rahaa > 100) {  
    ostaLevy();  
    rahaa -= levynHinta;  
}
```

# do-while - syntaksi

---

```
do {  
    lauseita;  
} while(ehto);
```

# do-while - toiminta

---

```
do {  
    System.out.println("Anna joku luku väliltä 1-9");  
    luku = lukija.nextInt();  
} while (luku < 1 || luku > 9);
```

# break

---

- Komennolla `break;` silmukan voi lopettaa kesken kaiken

- Esim:

```
while(kori ssaTi l aa) {  
    poi mi Mansi kka();  
    if(sataa) {  
        hei taMansi kkaMäkeen();  
        break; //l ähdet ään hi maan  
    }  
    asetaMansi kkaKori i n();  
}
```

# continue

---

- Komennolla `continue;` silmukan siirtyy suoraan seuraavalle kierrokselle
- Esim:

```
while(korissaTilaa) {  
    toimiMansiKka();  
    if(mansiKkaOnHuono) {  
        heiMansiKkaMäkeen();  
        continue; //alotetaan kädenliike alusta  
    }  
    asetaMansiKkaKoriin();  
}
```

# Lyhyesti

---

## Alkuehtoinen while

```
while (laskuri < 10) {  
    System.out.println("Hoi");  
    laskuri++;  
}
```

## Loppuehtoinen while

```
do {  
    System.out.println("Hei maailma!");  
    laskuri++;  
} while (laskuri < 10);
```