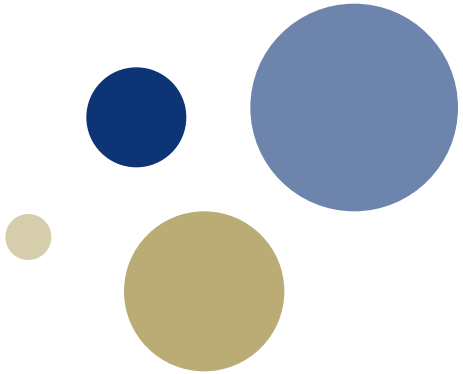
NTNU

Kunnskap for en bedre verden



Python: Valg og betingelser

TDT4110 IT Grunnkurs

Professor Guttorm Sindre

Læringsmål og pensum

- Mål
  - Kunne forstå og bruke if-setninger
  - … sammenlikning av strenger
  - … nøstede beslutningsstrukturer
  - … betingelser og uttrykk med logiske operatorer
  - … boolske variable
- Pensum
  - Starting out with Python: Chapter 4 / Chapter 3

  Decision Structures and Boolean Logic

if-setningen

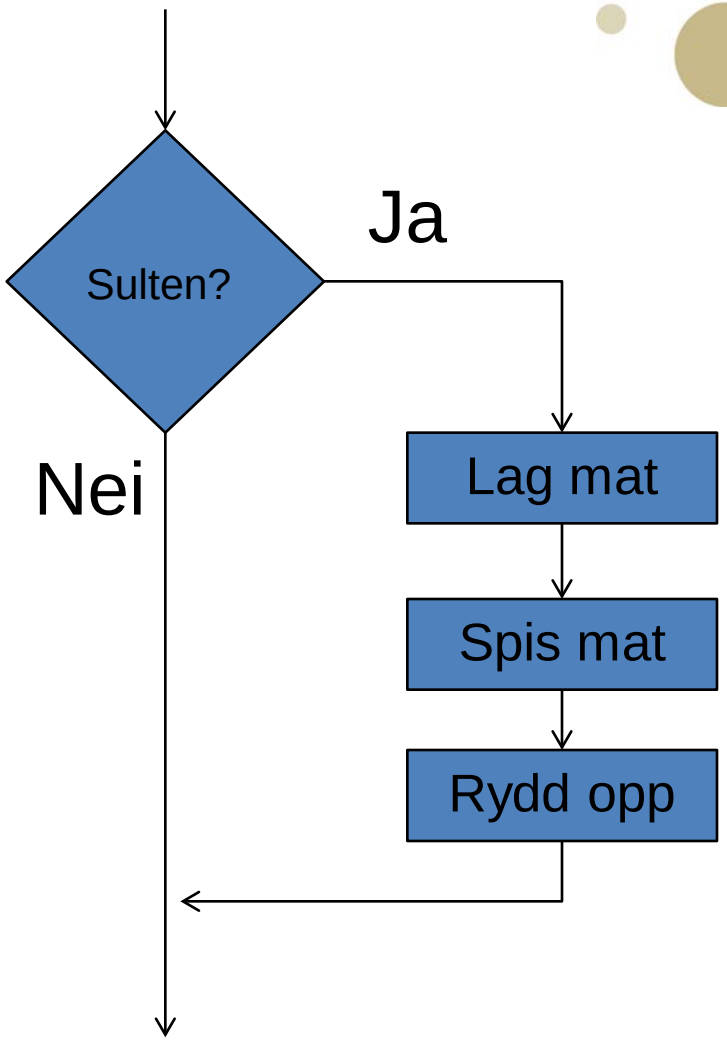
Kapittel 3.1

if-setninger

- HVORFOR trenger vi dette:
  - Ta beslutninger
  - Situasjonsbetingede handlinger
- HVORDAN virker if-setninger
  - Hvis en betingelse er tilfredsstilt, utføres handling (en eller flere kodelinjer)
  - Ellers utføres den ikke
  - Fortsetter deretter med kode som står etter if-setningen
  - **INNRYKK** viser hva som er del av if-setningen og hvor den slutter
- Syntaks:

```
if betingelse:
    kodelinje
    kodelinje
    etc.
```

Flytskjema



Betingelser i Python

- *relasjonsoperatorer* brukes ofte i betingelser
  - A la de *aritmetiske* operatorene +, -, *, /
  - Kan sammenligne
    - to tall (eller aritmetiske uttrykk som resulterer i tall)
    - tegn, tekster, boolske variable, etc.
- NB: ”er lik” i Python: ==
  - (mens = betyr tilordning)

Python	Matematikk	Forklaring
< >	< >	Mindre enn Større enn
<= >=	≤ ≥	Mindre eller lik Større eller lik
==	=	Er lik
!=	≠	Er ulik

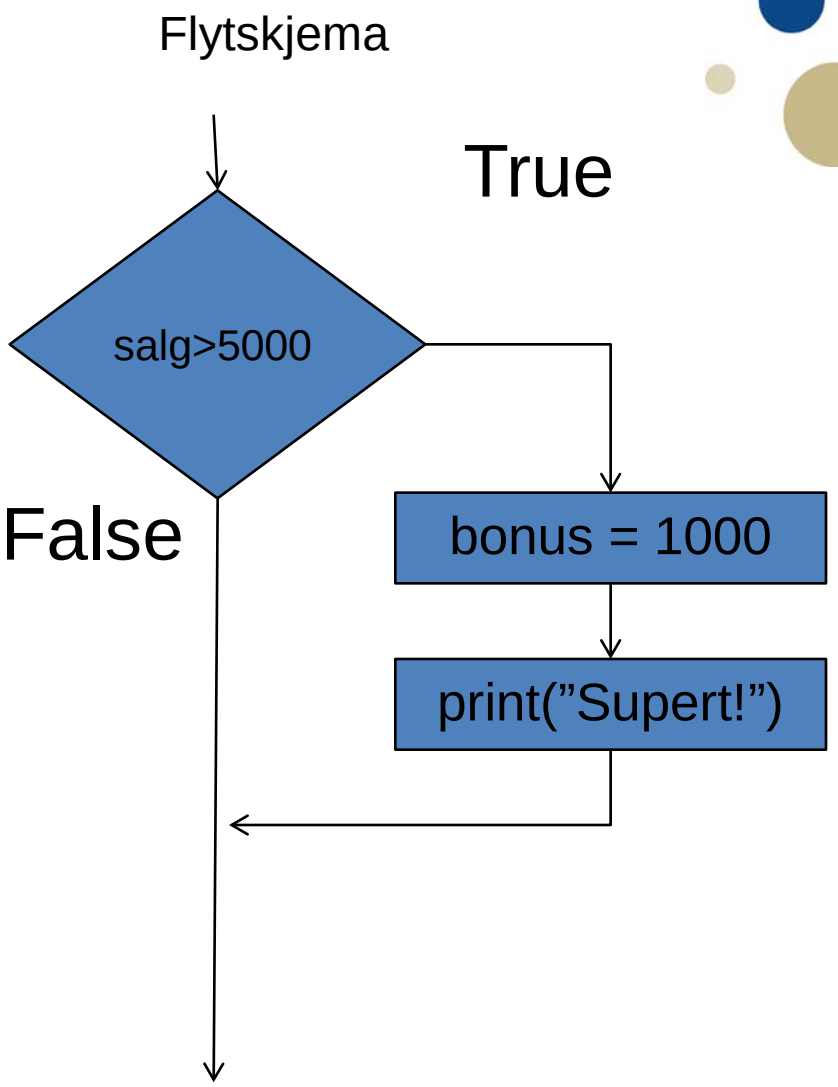
Eksempler på betingelser

Betingelse	Verdi
4 < 3	usann (False)
4==4	sann (True)
3!=3	usann (False)
3 > 3	usann (False)
3>=3	sann (True)

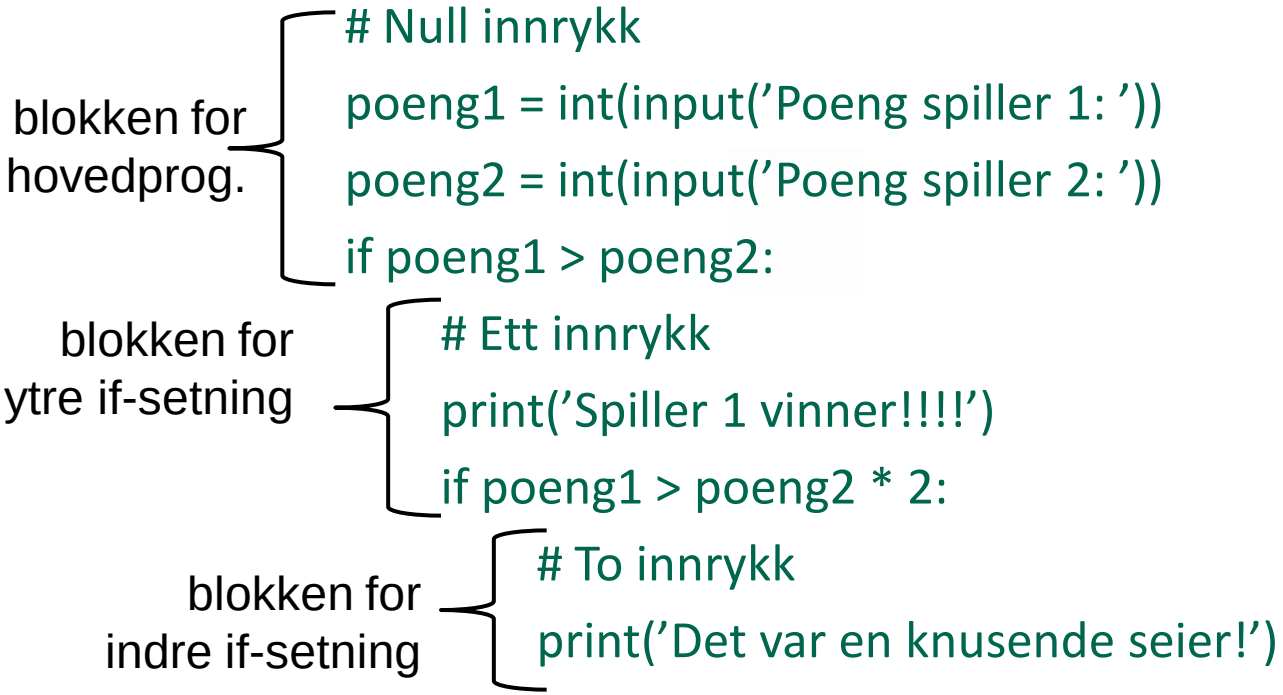
If-setningen

- Eksempel på if-setning:

```
if salg>5000:
    bonus = 1000
    print('Supert!')
```



Nøstede blokker



if-else uttrykk

Kapittel 3.2 / 4.2

if-else uttrykk

- Et if-else uttrykk vil
  - kjøre en blokk av kode hvis betingelsen er sann (True)
  - og en annen blokk av kode hvis betingelsen er usann (False).
- Velge mellom to alternative handlinger / kodeblokker
- Altså:
  - Det vi skal gjøre hvis betingelsen er sann:
    - puttes under if…
  - Det vi skal gjøre hvis betingelsen er usann:
    - puttes under else…
  - Det vi skal gjøre uansett om betingelsen er sann eller usann
    - Gjøres enten FØR hele if-else-setningen starter
    - Eller ETTER at hele if-else-setningen er slutt

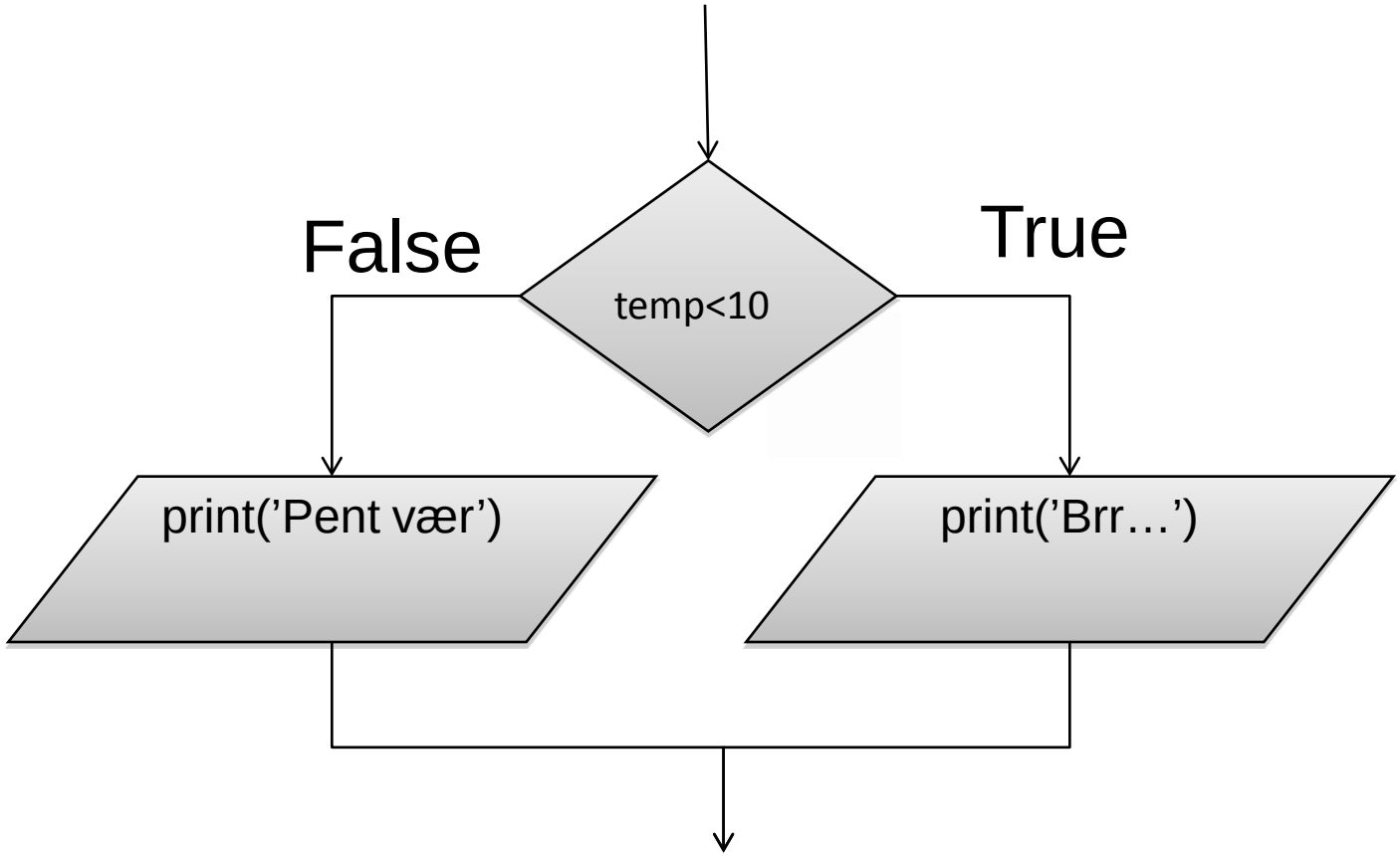
if-else generell kode

```
if betingelse:
    kode
    kode
    etc.
else:
    kode
    kode
    etc.
```

Denne kodeblokka blir utført hvis betingelsen er **sann**

Denne kodeblokka blir utført hvis betingelsen er **usann**

if-else flytskjema



if-else kodeeksempel

```
temp = int(input('Temperaturen er :'))
if temp<10:
    # Betingelsen er sann
    # Innrykk for kodeblokka
    print('Brrr!')
else:
    # Betingelsen er usann
    print('Pent vær')
```

Oppgave: if…else

- Skriv et program som sier om et år var vanlig eller skuddår:
  - INPUT: årstallet
  - OUTPUT: om det er vanlig eller skuddår
  - Bruk if-else-setning for avgjørelsen
- Enkel start: Reglene som gjaldt 8 e.Kr – 1582:
    - *Et år er skuddår hvis årstallet er delelig på 4*
      - *Eks.: 800 og 1580 var skuddår, 1002 og 1581 ikke*
- HINT: for å sjekke om tall går opp, bruk modulo
  - Eks.: **tall % 4 == 0** gir True hvis tallet er delelig på 4

Sammenlikne strenger

Kapittel 3.3 / 4.3

Sammenlikning av to variabler som inneholder strenger

- Variabler som inneholder tekststrenger kan sammenlignes på lik linje med tall.
- Eksempel på å sjekke om to variabler er like:

```
navn1 = ’Peter’
navn2 = ’Pelle’
if navn1==navn2:
        print(’Samme navn!’)
else:
        print(’Forskjellige navn’)
```

Sammenlikne variabel og streng

- Sjekke om en variabel inneholder en tekst:

```
passord = input('Skriv inn passord: ')

if passord == 'Nuff':
    print('Riktig passord')
else:
    print('Feil passord')
```

Sjekke om en streng er større enn en annen streng

- I Python kan du også sjekke om en streng er større (eller mindre) enn en annen streng.
  - Dvs. at en tekststreng har tegn som er representert med mindre eller større verdier enn i den andre strengen.
  - Alle tegn i Python representerer en tallverdi

```
if 'A' < 'B':
    print('Bokstaven A er mindre enn bokstaven B')
    # Bokstaven A representeres som 65, B som 66
```

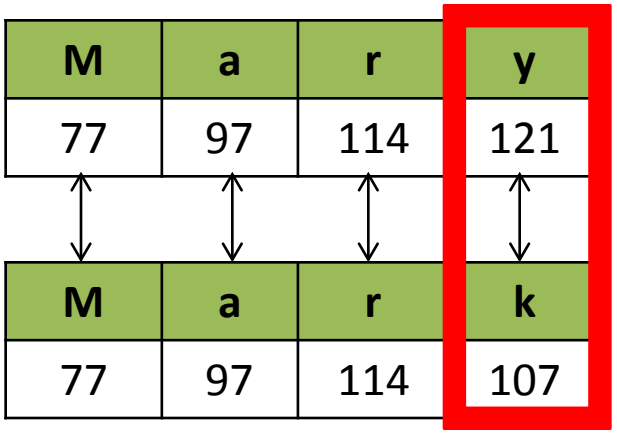
- Dette vil ofte funke for alfabetisk sammenligning
  - men ikke alltid

ASCII tabellen – tegn representert som tall

1 ┌	33 !	65 A	97 a	129 ▯	161 ¡	193 Á	225 á
2 ┐	34 "	66 B	98 b	130 ‚	162 ¢	194 Â	226 â
3 └	35 #	67 C	99 c	131 ƒ	163 £	195 Ă	227 ă
4 ┘	36 $	68 D	100 d	132 „	164 ¤	196 Ä	228 ä
5 │	37 %	69 E	101 e	133 …	165 ¥	197 Å	229 å
6 ─	38 &	70 F	102 f	134 †	166 ¦	198 Æ	230 æ
7 •	39 '	71 G	103 g	135 ‡	167 §	199 Ç	231 ç
8 ◘	40 (	72 H	104 h	136 ˆ	168 ¨	200 È	232 è
9	41)	73 I	105 i	137 ‰	169 ©	201 É	233 é
10	42 *	74 J	106 j	138 Š	170 ª	202 Ê	234 ê
11 ♂	43 +	75 K	107 k	139 ‹	171 «	203 Ë	235 ë
12 □	44 ,	76 L	108 l	140 Œ	172 ¬	204 Ì	236 ì
13	45 -	77 M	109 m	141 ▯	173 -	205 Í	237 í
14 ♫	46 .	78 N	110 n	142 Ž	174 ®	206 Î	238 î
15 ☼	47 /	79 O	111 o	143 ▯	175 ¯	207 Ï	239 ï
16 ┼	48 0	80 P	112 p	144 ▯	176 °	208 Đ	240 ð
17 ◄	49 1	81 Q	113 q	145 '	177 ±	209 Ñ	241 ñ
18 ↕	50 2	82 R	114 r	146 '	178 ²	210 Ò	242 ò
19 ‼	51 3	83 S	115 s	147 "	179 ³	211 Ó	243 ó
20 ¶	52 4	84 T	116 t	148 "	180 ´	212 Ô	244 ô
21 ┴	53 5	85 U	117 u	149 •	181 µ	213 Ő	245 ő
22 ┬	54 6	86 V	118 v	150 –	182 ¶	214 Ö	246 ö
23 ┤	55 7	87 W	119 w	151 —	183 ·	215 ×	247 ÷
24 ↑	56 8	88 X	120 x	152 ˜	184 ¸	216 Ø	248 ø
25 ├	57 9	89 Y	121 y	153 ™	185 ¹	217 Ù	249 ù
26 →	58 :	90 Z	122 z	154 š	186 º	218 Ú	250 ú
27 ←	59 ;	91 [	123 {	155 ›	187 »	219 Û	251 û
28	60 <	92 \	124 |	156 œ	188 ¼	220 Ü	252 ü
29	61 =	93]	125 }	157 ▯	189 ½	221 Ý	253 ý
30	62 >	94 ^	126 ~	158 ž	190 ¾	222 Þ	254 þ
31	63 ?	95 _	127 ⌂	159 Ÿ	191 ¿	223 ß	255 ÿ
32	64 @	96 `	128 €	160	192 À	224 à	

Sammenlikning av to strenger

- Hva skjer her? Hva sammenliknes?
  - Sjekker bokstav for bokstav!



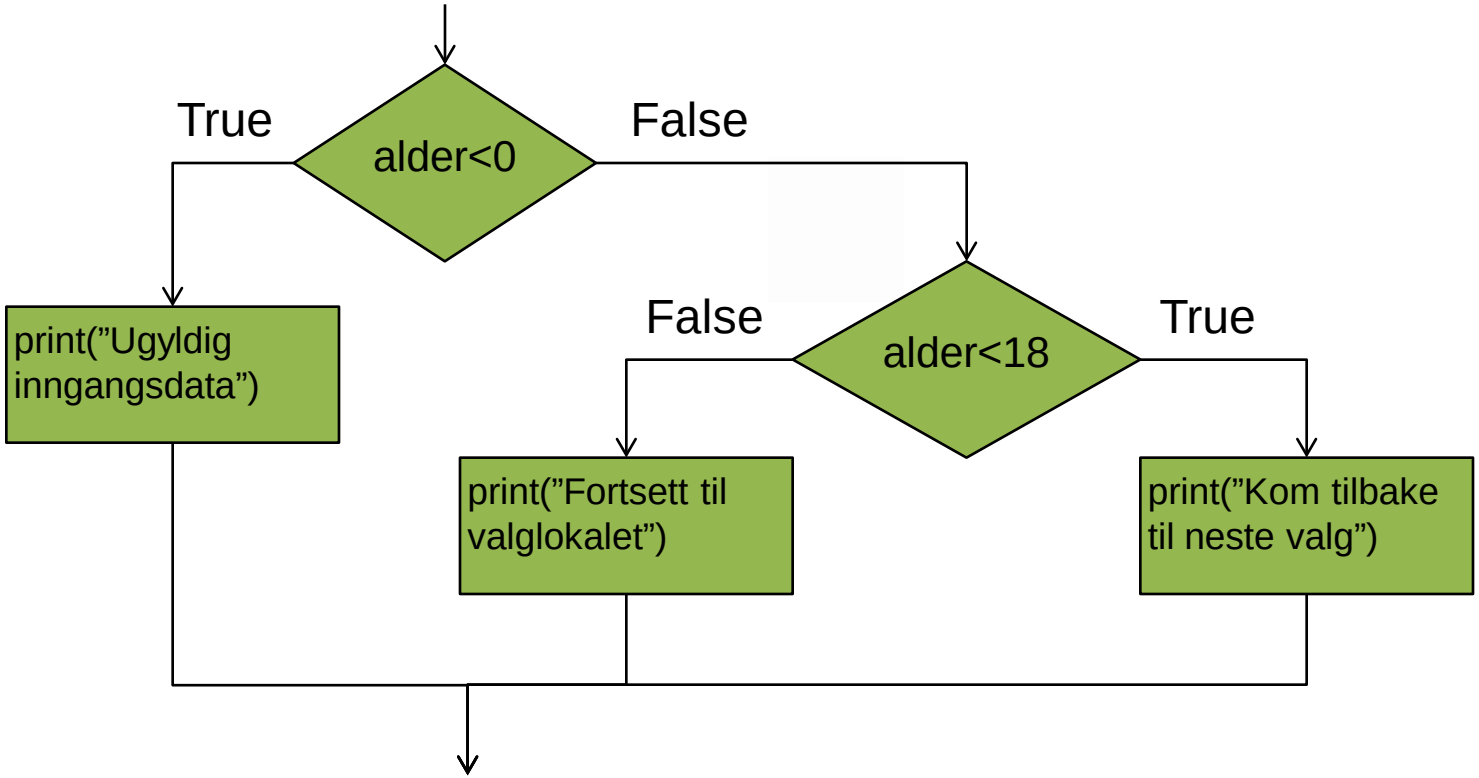

```
navn1 = 'Mary'
navn2 = 'Mark'
if navn1 > navn2:
    print('Mary er alfabetisk etter Mark')
else:
    print('Mary er alfabetisk før Mark')
```

Nøstede betingelser og if-elif-else

Kapittel 3.4 / 4.4

Flytskjema for nøstede if-setninger

- Vi kan skrive flere if-setninger inne i hverandre (nøsting)



Nøsting av if-setninger

- Kodeeksempel på *nøstede* if-setninger:

```
if (alder <0):
    print(”Ugyldige inngangsdata”)
else:
    if (alder <18):
        print(”Kom tilbake til neste valg”)
    else:
        print(”Forsett til valglokalet”)
```

- *innrykk* viser at de indre setningene hører til if-setningen.
- En if-setning avsluttes ved å fjerne innrykk.
- Kan ha flere nivåer med if-setninger inni hverandre.

Bruk av if-elif-else

- Nøsting av setninger kan fort bli uoversiktlig
- Python har derfor elif for bedre lesbarhet.
  - elif er en forkortelse for **else if** (hvis ikke det ovenfor slår til, så….)

```
if (poeng>=89):
    karakter = ’A’
elif (poeng>=77):
    karakter = ’B’
elif (poeng>=65):
    karakter = ’C’
elif (poeng>=53):
    karakter = ’D’
elif (poeng>=41):
    karakter = ’E’
else:
    karakter = ’F’
```

- NB! Kun en av betingelsene vil slå til!

Oppgave: skuddår

- Endre programmet for skuddår
  - I stedet bruke nåværende regler for skuddår (1582-):
    - *Et år er skuddår hvis årstallet er delelig på 4*
      - *Eks.: 2012 var skuddår, 2014 er ikke*
    - *…men likevel IKKE skuddår hvis delelig på 100*
      - *Eks.: 1900 var ikke skuddår*
    - *…men likevel skuddår hvis det også er delelig på 400*
      - *Eks.: 2000 var skuddår*
  - Lag kode som tester dette riktig
    - Bruk nøstede if-else-setninger eller if-elif-…-else

Logiske operatorer

Kapittel 3.5

Logiske uttrykk

- På samme vis som vi har sammensatte *aritmetiske* uttrykk kan vi sette sammen betingelser til vilkårlig store uttrykk
- Dette kaller vi *logiske uttrykk*
- Vi kaller ”limet” som binder disse sammen for *logiske operatorer*
- Python definerer de følgende logiske operatorene slik:

Operator i Python	Forklaring
and	Logisk og
or	Logisk eller
not	Logisk ikke, eller negasjon

Logiske uttrykk (fortsettelse)

- Hva betyr **`and`**, **`or`** og **`not`** i praksis:
  - **`and`** gir True kun hvis begge sider er True:

    `False and True` gir False

    `False and False` gir False

    `True and True` gir True
  - **`or`** gir True hvis minst en av sidene er True:

    `False or True` gir True

    `True or True` gir True

    `False or False` gir False
  - **`not`** gir motsatt sannhetsverdi av den vi hadde:

    `not True` gir False

    `not False` gir True

Eksempel på logiske uttrykk

- Vanlig bruk er å sjekke at en verdi ligger i et intervall:

```
x >= 5 and x <= 10
```

- kan også sjekke intervaller på følgende måte:

```
5<= x <= 10
```

- Parenteser for større uttrykk med flere betingelser:

```
(i>=1 and i<=N) or (j>=1 and j<=N)
```

Resultatet av logiske utregninger

- En enkel eller sammensatt betingelse kalles et *logisk uttrykk*
- Evaluering av et logisk uttrykk gir enten `True` eller `False`
  - … men IKKE sann eller usann!

For å få riktige betingelser husk presedens-rekkefølgen:

1. ( )
2. ** # Eksponent (opphøyd)
3. *, /, //, % # heltallsdivisjon, rest
4. +, -
5. <, <=, >, >=, <>, !=, ==
6. not
7. and
8. or
9. if – else
10. Lik prioritet: fra venstre mot høyre

- Bruk parenteser hvis nødvendig
  - De evalueres innenfra og ut

Betingelser

- Oppgave: Er denne betingelsen **True** eller **False**?
  - `4<7 and not(3>1 or 8>=9)`

- Begynn innenfra og jobb utover (inne i parenteser)

En liten test...
Bestem true eller false

Anta at variablene har fått verdier som følger:

A = 5, B = 9, C = 12, D = 39

Hva blir den boolske verdien av uttrykkene?

- (B<=C)
- (A>5 or B==2)
- (A+B < C+D) and (D>=39)
- (A>B or B>C or C>D or D>A)

Boolske variable

Kapittel 4.6

Boolske variable

- En boolsk variabel kan referere til en av to verdier:
  - True eller False.
- Kan lagre at en spesiell betingelse er sann eller ikke.

```
aar=int(input('Skriv inn årstall: '))
skuddaar = aar % 4 == 0          #blir True/False
# N linjer lenger nede i programmet…
if skuddaar:
        antall_dager=366
# ytterligere M linjer lenger nede i programmet…
if skuddaar:
        antall_d_feb=29
else:
        antall_d_feb=28
```

Avslutning if-setninger

- Unngå overflødig bruk av `not` – tungt å lese
- Ved if else,
  - skriv helst positiv utfall i if og negativt i else
- Vi kan ha flere setninger mellom if … else
  - Innrykk avgjør hvor koden hører til
- Typiske feil man kan gjøre
  - Noen setninger kan umulig slå til
    - feil betingelse, eller feil rekkefølge på betingelser
  - Mer enn en setning slår til
    - brukt flere if hvor det skulle vært elif?

Oppgave: Skuddår igjen

- Gjør programmet for skuddår kortere
  - Bruk **and, or, not**
    - Kan dermed klare oss med en enkel if-else-setning
    - Unngå nøsting eller bruk av **elif**

Oppsummering

- Betingelser i Python: < , > , <= , >= , == , !=, <>
- Operatorer for logiske uttrykk: and, or , not
- Logiske uttrykk kan enten bli False eller True
- if-setninger:

```
if (<betingelse>):
    <utfør noe>          # HUSK INNRYKK!
elif (<betingelse):
    <utfør noe>          # HUSK INNRYKK!
else:
    <utfør noe annet>
```

- Vi kan også bruke nøstede if-setninger
- Innrykk er avgjørende for logikken i programmet

Oppgave: Mer skuddår

- Ekstraoppgave for spesielt interesserte
  - Trene på å lage en stor, sammensatt if-struktur
- Lag et skuddårsprogram med komplette regler
  - INPUT: Årstallet (negativt tall hvis f.Kr.)
  - OUTPUT: Om året var/er skuddår eller ikke
    - Skriv ’f.Kr.’ og ’e.Kr.’ i stedet for negative og positive tall
- Regler for skuddår:
  - T.o.m. 46 f.Kr.: Ingen skuddår
  - 45 f.Kr. – 9 f.Kr.: Skuddår hvis delelig på 3
  - 8 f.Kr. – 7 e.Kr.: Ingen skuddår (pause)
  - 8.e.Kr. – 1581: Skuddår hvis delelig på 4
  - 1582 – …: delelig på 4 men ikke på 100 (unntatt delelig 400)

Neste uke: Løkker (kap.4)

- Aktuelle spørsmål for quiz:
  - «How many times …» (Checkpoint 4.6)
  - «Rewrite the following code…» (Checkpoint 4.8)
  - «What will the following code display?» (Checkpoint 4.12)
  - «What will the following code display?» (Checkpoint 4.15)
  - «Why should you take care … sentinel?» (Checkpoint 4.19)
  - «Describe the steps that are generally…» (Checkpoint 4.22)
  - «Which of the following…» (Review Questions Multiple Choice 2)
  - «Each repetition of a loop is known as a(n) ____» (Review, MC 3)
  - «_____ is a keyword that is used…» (Review Questions, MC 12)
  - «In a while loop the keyword continue…» (Review, True or False 3)
  - «Why is it critical that accumulator variables…» (Review, Short Answer 4)
  - «Write a for loop that displays the following set of numbers…» (Algorithm Workbench 3)
  - «Write code that prompts the user to enter…» (Algorithm Workbench 8)
- Noen av disse gis i «kahootisert form»
- Pluss 1-2 helt uannonserte spørsmål, men også om løkker