IN1010 Gruppetime Uke 0

Gruppe 2 - Design

Om meg

- Jeg heter Timothy
- Går Prosa, 3. året
- Har vært gruppelærer i:
  - IN1000
  - IN1010 (Vår 2025)
  - IN2010
- Fra Bamble i Telemark
- Musikknerd
- Fun fact: Favorittalbumet mitt er Imaginal Disk – Magdalena Bay





Navnerunde

- Navnerunde i plenum
- Si
  - Navnet ditt
  - Linja di
  - Favorittsangen din akkurat nå
- (Jeg er vilt dårlig på navn, så jeg kan ikke love at jeg husker de)

Viktig informasjon

- Kontakt meg på epost: timothbj@uio.no
  - Kan ikke love at jeg kan svare på alle spørsmål utenom min arbeidstid :)
- Sjekk semestersida jevnlig
  - Jeg legger ut ressurser på gruppesida til gruppe 2
- Trix: https://trix.ifi.uio.no/course/16
- To eksamener: Én midtveis og en avsluttende
- Husk Discourse!
- 2 obligatoriske innleveringer
  - Lurt å gjøre alle innleveringene, selv de frivillige
- Innlevering 0 lagt ut (gjør den!)
- Beregn mer tid enn dere gjorde i IN1000

Mine erfaringer med IN1010

- Vanskeligere å sette seg inn i enn IN1000
  - Hadde erfaring med programmering i Python fra videregående, hadde aldri rørt Java før
- Man blir fort vant til de nye konvensjonene
- Gjorde en del feil i starten
  - Å glemme semikolon er noe jeg fortsatt gjør
- Kode kjører som oftest aldri første gang
  - Java krever mye pirk og feilsøking
- Jeg fikk god karakter av å
  - Kode **mye!**
  - Jobbe/diskutere med venner
  - Sette av mye tid til emnet

Eksamen

- Midtveiseksamen 20. mars kl 15:00 - 17:00
  - Ingen hjelpemidler tillatt
- Avsluttende eksamen 29. mai kl 15:00 - 19:00
  - Alle trykte og skrevne hjelpemidler
- Begge eksamenene er på Silurveien 2
- Midtveis teller 25%, avsluttende teller 75% (av endelig karakter)
- Det pleier å tilbys konte til både midtveis og avsluttende eksamen (kan ikke love at det blir konte i midtveis)

Litt om gruppetimene

- Oppfordrer til diskusjon
  - Det hjelper veldig mye å snakke med sidemannen
- Kommer nok til å ha fokus på problemløsning
- Skal prøve å livekode en del
- Ikke nøl med å stille spørsmål!
  - Ingen spørsmål er dumme

Java vs. Python

Alt foregår i klasser

- Hver Java-fil **må** ha en klasse
  - Klassen må hete det samme som filen
- Hvis man ikke definerer en klasse, får man ikke kjørt koden
- Selv en hovedprogram-fil må være en klasse

```
1  class Uke0:
2      # Noe her
3
```

Uke0.java

Uke 0 > Uke0.java > ...

```
1  class Uke0 {
2
3  }
4
```

Merk at Java bruker krøllparantes, ikke innrykk

Main-metode

- Siden alt foregår i klasser, må vi ha en egen metode for å kjøre kode
- All kode som ikke er i en main-metode kan ikke kjøres
- Definisjon av main-metoden er litt heavy i starten, men det kommer etter hvert
- Akkurat nå trenger dere ikke forstå hva alle ordene betyr

```
class Uke0 {
    Run main | Debug main | Run | Debug
    public static void main(String[] args) {
        // Kode her
    }

}
```

Dynamisk vs. Statisk typet

- Python er **dynamisk** typet, altså at variabler kan bli tilegnet hvem som helst type
- Java er **statisk**, man må spesifisere type på basically alt
  - Hvis man har spesifisert type på en variabel, kan variabelen bare referere til et objekt/verdi av samme type
  - Man trenger kun å spesifisere typen til variabelen når man oppretter den

Python

```python
tall = 47
flyttall = 19.9
sant = True
streng = "Null semikolon"

tall = "Ikke feilmelding!"

```

Java

```java
class Uke0 {
    Run main | Debug main | Run | Debug
    public static void main(String[] args) {
        int tall = 47;
        double flyttall = 19.9;
        boolean sant = true;
        String streng = "Husk semikolon!";

        tall = "Feilmelding!";
    }

}
```

Utskrift

- I Python skriver man ut med print()
- I Java skriver man ut med System.out.println()
- To versjoner:
  - System.out.println(): Newline bak utskrift
  - System.out.print(): Ikke newline bak utskrift

```java
class Uke0 {
    Run main | Debug main | Run | Debug
    public static void main(String[] args) {
        int tall = 47;
        double flyttall = 19.9;
        boolean sant = true;
        String streng = "Husk semikolon!";

        System.out.println(streng + tall);
    }

}
```

```
Husk semikolon!47
```

Hvordan kjøre et Java-program?

- Java kode må **kompileres** før det kan kjøres
  - Python bruker en interpreter som tolker koden under kjøring
  - Javakode må tolkes før kjøring
- Her **må** dere bruke terminalen
- Viktige kommandoer:
  - javac Fil.java – Kompilerer filen til en .class fil
  - java Fil – Kjører filen (merk at man ikke trenger .class her)

```
14:48 timothbj@MrMorale/Uke 0: javac Uke0.java
14:48 timothbj@MrMorale/Uke 0: java Uke0
Husk semikolon!47
14:48 timothbj@MrMorale/Uke 0:
```

Husk å kompilere på nytt dersom du har gjort en endring!

(Og vil kjøre programmet med den nye endringen)

La oss lage et program!

Hello world

If-setninger

- If-setninger og blokker ligner veldig på Python
  - Hovedforskjellen er krøllparanteser og parantes rundt sannhetsutrykket
- elif -> else if
- Ellers er det ganske likt
- Husk at sannhetsuttrykket inne i parantesen må resultere i en verdi av typen 'boolean'

Python

```python
verdi = 54

if verdi > 100:
    print("Wow!")
elif verdi > 50:
    print("Tja...")
else:
    print("Nuh uh")
```

Java

```java
class Uke0 {
    Run main | Debug main | Run | Debug
    public static void main(String[] args) {
        int verdi = 54;

        if (verdi > 100) {
            System.out.println("Wow!");
        } else if (verdi > 50) {
            System.out.println("Tja...");
        } else {
            System.out.println("Nuh uh");
        }
    }

}
```

Oppgave

- Lag en Java fil som heter Klubben.java
- Lag en int ‘alder’ som du bestemmer verdi på
- Lag en if-blokk som printer ut
  - ‘For gammel’ hvis du er 40 eller eldre
  - ‘Kom inn!’ hvis du er yngre enn 40 men eldre eller lik 18
  - ‘Dra tilbake til barnehagen’ hvis du er yngre enn 18

Løkker i Java

- While-løkker i Java er ganske likt som i Python
  - Igjen, hovedforskjell er krøllparantes og parantes rundt sannhetsuttrykket
- For løkker er litt annerledes
  - for i in range() er ikke en greie lenger
- For-løkker krever at man spesifiserer når løkka starter, og når løkka stopper

While

Python

```python
verdi = 12

while verdi < 20:
    verdi += 1
    print(verdi)
```

Java

```java
class Uke0 {
    Run main | Debug main | Run | Debug
    public static void main(String[] args) {
        int verdi = 12;

        while (verdi < 20) {
            verdi++;
            System.out.println(verdi);
        }
    }

}
```

For

Python

```python
for i in range(30):
    print(i)
```

Java

```java
class Uke0 {
    Run main | Debug main | Run | Debug
    public static void main(String[] args) {
        for (int i = 0; i < 30; i++) {
            System.out.println(i);
        }
    }

}
```

Løkka går helt til i ikke er mindre enn 30



i defineres som 0

i øker med 1 hver iterasjon

Oppgave

- Lag et program som heter Bryggeri.java
- Bruk en for-løkke for å printe ut ‘100 flasker øl på hylla’, så ’99 flasker øl på hylla’, osv, helt fram til ‘0 flasker øl på hylla’
- Til slutt, kompiler og kjør programmet og pass på at dere får riktig utskrift
- Hint: Hva er det motsatte av i++?

	Python	**Java**
Koden	tolkes direkte	kompileres til `.class`-fil som kjøres
Kode som skal kjøres må legges	i en fil	i main-metode i en klasse
Kodeblokker markeres med	innrykk	{ og }
Kommandoer separeres med	linjeskift	semikolon
Variabler/funksjoner/metoder er	utypet	typet
Kommentar starter med	#	//
Ingen verdi	`None`	`null`

If-test	`if x < 2:`
	`  ...`
	`elif x > 2:`
	`  ...`
	`else:`
	`  ...`
		`if (x < 2) {`
		`  ...`
		`} else if (x > 2) {`
		`  ...`
		`} else {`
		`  ...`
		`}`

While-løkke	`x = 0`
	`while x < 10:`
	`  ...`
	`  x += 1`
		`int x = 0;`
		`while (x < 10) {`
		`  ...`
		`  x++;`
		`}`

For-løkke	`for x in range(0, 10):`
	`  ...`
		`for (int x = 0; x < 10; x++) {`
		`  ...`
		`}`

Spørsmål?

Trix/Lab