

Vi starter igjen 09:15

Forelesningen filmes

Forelesningen starter 08:15

Forelesningen filmes

Øvingsforelesning 0

Intro til Java

Eirik Lorgen Tanberg

Vitenskapelig Assistent ITGK

Agenda

- Administrativt og tips til VSCode
- Gjennomgang av oppgaver i øving 0
- Lære hvordan lister og dictionaries fungerer i Java
- Gjøre oppgaver fra ITGK, men skrive i Java

Målet i dag er å bli mer komfortabel med Java, slik at dere er i bedre stand til å anvende Java til OOP.

Hjelpemidler i VSCode

- VSCode har mange triks og hurtigtaster man burde lære seg for å effektivisere prosessen.
- Man kan lage **snippets** som genererer kode for deg!
- Navigere i kode – kan f.eks. åpne/lukke klasser/metoder, mm. for å få bedre oversikt
- Kan enkelte hente opp **metodedeklarasjon** (CTRL+museklikk (win) eller CMD+museklikk (mac) på koden vi vil finne)

Noen nyttige kommandoer/snarveier å lære seg:

- **CMD/CTRL+F**: Søk opp et tegn/ord i koden
- **CMD/CTRL+Shift+F**: Søk opp et tegn/ord i **alle filer i prosjektet**
- **CMD/CTRL+P**: Søk etter **navn på filer**
- **CMD/CTRL+Shift+P**: Åpne **command palette** hvor du lett kan gjøre en drøss med ting
- **ALT+Shift+Pil opp/ned**: Kopier nåværende linje opp eller ned
- **CMD/CTRL+N**: Åpne en ny **tom fil**

Øvingsgrupper

- Valg av læringsassistent gjøres på Blackboard. Gå inn på emne-forsiden → **Grupper** → **Vis påmeldingsskjema for å bli med i en gruppe** → Finn gruppe og trykk **Meld deg på**
- Påmelding åpner klokken (ca) **12:00 i dag, 17.01.2024**
- Valgt feil gruppe? Send e-post til undassene på tdt4100-undass@idi.ntnu.no

Mappestruktur for ØF

foreksempel/src/main/java

of0

kode

← Her kan du kode direkte selv

lf

← Løsningsforslag, legges ut i etterkant

of1

.

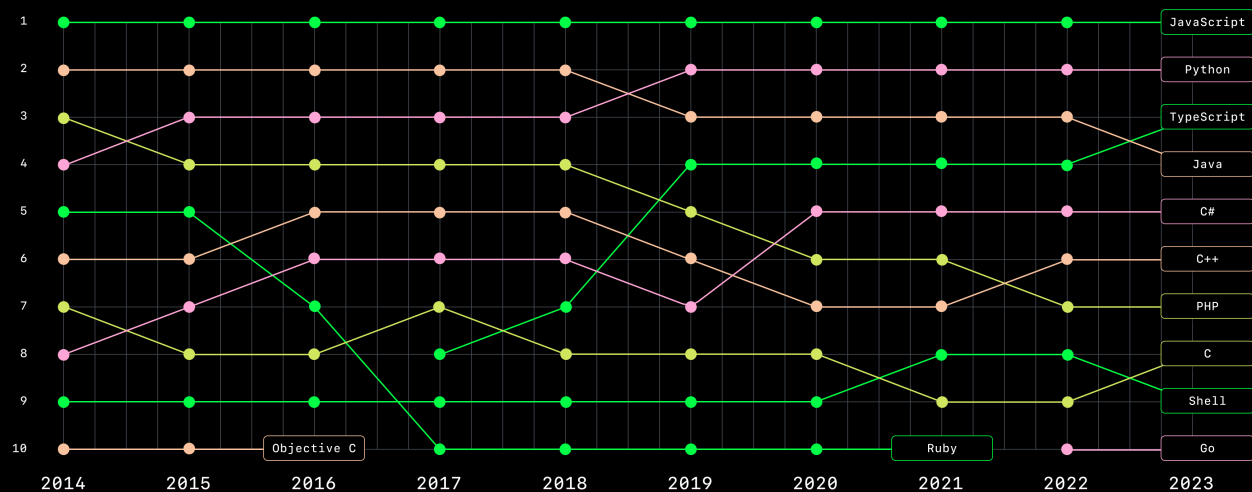
.

.

of13

Hvorfor Java?

Top 10 programming languages on GitHub



<https://github.blog/2023-11-08-the-state-of-open-source-and-ai/>



Back-end (Server-side) table in most popular websites

Websites ↕	C# ↕	C ↕	C++ ↕	D ↕	Elixir ↕	Erlang ↕	Go ↕	Hack ↕	Haskell ↕	Java ↕	JavaScript ↕	Perl ↕	PHP ↕	Python ↕	Ruby ↕	Scala ↕	XHP ↕
Google	No	Yes	Yes	No	No	No	Yes	No	No	Yes	Yes	No	No	Yes	No	No	No
YouTube	No	Yes	Yes	No	No	No	Yes	No	No	Yes	No	No	No	Yes	No	No	No
Facebook	No	No	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	No	No	Yes	Yes	No	No	Yes
Yahoo	No	Yes	Yes	No	No	No	Yes	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
Amazon	No	No	Yes	No	No	No	No	No	No	Yes	No	Yes	No	No	No	No	No
Wikipedia	No	No	No	No	No	No	No	No	No	No	No	No	Yes	No	No	No	No
Twitter	No	No	Yes	No	No	No	No	No	No	Yes	No	No	No	No	Yes	Yes	No
Bing	Yes	No	Yes	No	No	No	No	No	No	No	No	No	No	No	No	No	No
eBay	No	No	No	No	No	No	No	No	No	Yes	Yes	No	No	No	No	Yes	No
MSN	Yes	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No	No
LinkedIn	No	No	No	No	No	No	No	No	No	Yes	Yes	No	No	No	No	Yes	No
Pinterest	No	No	No	No	Yes	Yes	No	No	No	No	No	No	No	Yes	No	No	No
WordPress.com	No	No	No	No	No	No	No	No	No	No	No	No	Yes	No	No	No	No
Netflix	No	Yes	No	No	No	No	Yes	No	No	Yes	No	No	No	Yes	No	No	No



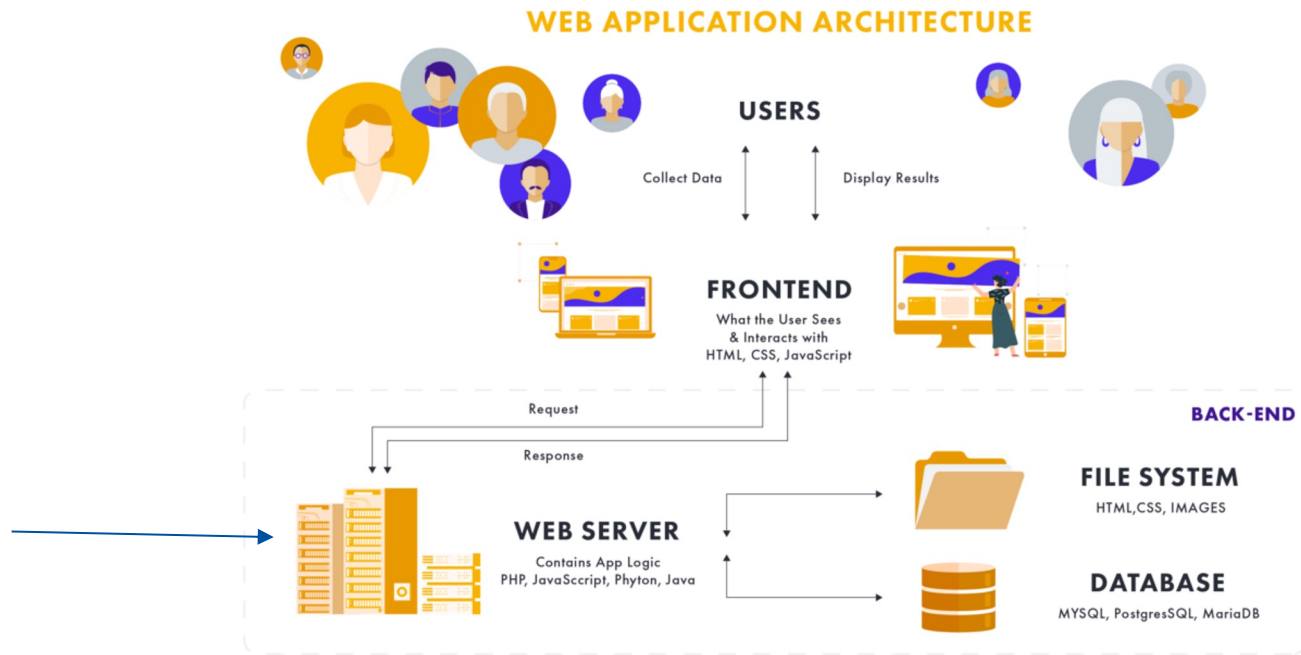
https://en.wikipedia.org/wiki/Programming_languages_used_in_most_popular_websites



android



Java sin plass i moderne webapplikasjoner



Øving 0

- Vi gjør “Python VS Java”-oppgavene i øving 0

Python VS Java

Noe av det første du kommer til å merke når du skal programmere i Java er at det kreves en del flere linjer kode enn i Python. La oss starte med noe av det enkleste, å skrive ut en tekst til konsollen.

Skrive ut tekst til konsollen

I Python er det veldig greit å skrive ut tekst til konsollen. Du trenger bare å skrive `print("tekst")` og så vil teksten skrives ut til konsollen. I Java trengs det litt bokstaver:

Python

```
print("Hello World!")
```

Java

```
System.out.println("Hello World!");
```

En snarvei i VS Code for å slippe å skrive hele `System.out.println()` er å skrive `sout` og trykke `tab`-tasten. Da vil hele koden bli skrevet ut for deg.

Variabler

I Python er det ganske rett frem å lage variabler. Du trenger bare å skrive `variabelnavn = verdi`. I Java må du først angi hvilken type variabelen skal være. De vanligste typene er `int`, `double`, `boolean` og `String`. `double` er et desimaltall (samme som `float` i python), `boolean` er en `true` eller `false`-verdi og `String` er en tekststreng. La oss se på et eksempel:

Python

```
x = 5
y = 10.6
s = "hei"
ja = True
```

Java

```
int x = 5;
double y = 10.6;
String s = "hei";
boolean ja = true;
```

Oppgave 1: Print

OPPGAVE

- Lag to variabler som inneholder tallene du skal gange sammen. Du kan kalle dem x og y eller noe annet du vil. Velg selv om du vil bruke int eller double.
- Lag en variabel som inneholder resultatet av gangeoperasjonen. Du kan kalle den z eller noe annet du vil.
- Skriv ut resultatet til konsollen ved å bruke `System.out.println();`.

Oppgave 2: If

OPPGAVE

- Oversett python-koden under til java:

```
x = 3
y = 5
if x > 5 and y < 10:
    print("x er større enn 5 og y er mindre enn 10")
elif x > 5 or y < 10:
    print("x er større enn 5 eller y er mindre enn 10")
else:
    print("x er mindre enn 5 og y er større enn 10")
```

Oppgave 3: Løkker

OPPGAVE

- Oversett python-koden under til java:

```
for i in range(3,10):  
    if i % 2 == 0:  
        print(i)  
  
j = 0  
while j < 10:  
    print(j)  
    j += 1
```

Oppgave 4: Funksjoner

OPPGAVE

- Oversett python-koden til java:

```
def division(x, y):  
    return x / y  
  
def faktet(x):  
    fak = 1  
    for i in range(1, x+1):  
        fak *= i  
    return fak  
  
def erPrimtall(x):  
    if x < 2:  
        return False  
    for i in range(2, x):  
        if x % i == 0:  
            return False  
    return True
```

Kort om “input” i Java

- Vanligvis lite brukt i større programmer
 - Man bruker som regel et grafisk brukergrensesnitt, ikke terminal
 - Enklest å bruke “Scanner”-klassen
 - Fungerer ikke i “Debug”-terminalen i VS Code
 - Kommentér ut linje 3 i vscode/settings.json



```
x = input("Skriv inn et tall: ")  
  
print(x)
```

```
import java.util.Scanner; // Import the Scanner class  
  
class Main {  
    public static void main(String[] args) {  
        Scanner myObj = new Scanner(System.in); // Create a Scanner object  
        System.out.println("Skriv inn et tall: ");  
  
        String tall = myObj.nextLine(); // Read user input  
        System.out.println(tall); // Output user input  
    }  
}
```



Input i VSCode terminal

Fjern denne linjen for at input gjennom terminal skal fungere

```
.vscode > {..} settings.json > java.debug.settings.console  
1 {  
2   "java.configuration.updateBuildConfiguration": "automatic",  
3   "java.debug.settings.forceBuildBeforeLaunch": false,  
4   "java.debug.settings.console": "internalConsole"  
5 }
```

Oppgave 5: Input

OPPGAVE

- Be brukeren om et tall.
- Hvis ikke input er et heltall, skriv ut en passende feilmelding.
- Hvis tallet er et heltall, skriv ut tallet.



Lister i Java

- Vi har flere typer lister i Java.
 - Den som ligner mest på Python-lister heter **ArrayList**, som vi anbefaler å bruke.



```
min_liste = [1,2,3]

tom_liste = []

tom_liste.append(1)
```

```
import java.util.*;

public class ArrList {
    public static void main(String args[])
    {

        // create a ArrayList Integer type
        // and Initialize an ArrayList with List.of()
        ArrayList<Integer> minListe = new ArrayList<>(List.of(1, 2, 3));

        ArrayList<Integer> tomListe = new ArrayList<>();

        tomListe.add(1)

    }
}
```



Lister i Java

- Vi har flere typer lister i Java.
 - Den som ligner mest på Python-lister heter **ArrayList**, som vi anbefaler å bruke

index	0	1	2	3	4	5	6
element	'a'	'b'	2	'd'	‘!’	True	99

Oppgave 6: Lister

OPPGAVE

a)

- Lag en tom liste som skal inneholde strenger
- Legg til tre valgfrie elementer i listen med `add()`
- Hent elementet på index 1 og print det



Oppgave 6: Lister

OPPGAVE

b)

- Initialiser en ny liste som inneholder tallene 2, 1.2, og 5
- Print ut lengden av listen



Dictionaries i Java

- Det nærmeste vi kommer en Python-dictionary i Java er HashMap-klassen.
- **Viktig:** Vi må bestemme typen på både key og value

keys	"Norge"	"Sverige"	"Island"	"Italia"	"Spania"	"Finland"
values	"Oslo"	"Stockholm"	"Reykjavik"	"Roma"	"Madrid"	"Helsinki"

Oppgave 7: Dictionaries

OPPGAVE

- Lag en HashMap med verdier som vist til høyre
- Karl Popper ble født i 1902. Legg dette til i HashMap-en med “put”.
- Fjern “Hume” med “Remove”-metoden
- Print ut antall elementer i map-et

```
Filosof = {"Platon": -428,  
           "Hume": 1711,  
           "Aristoteles": -384,  
           "Descartes": 1596  
          }
```

ITGK-oppgaver

- Vi gjør noen oppgaver fra ITGK, men nå må vi skrive koden i Java
- Benytt filen Itgk.java

Øving 2 - Sammenligning av strenger*

OPPGAVE

- Lag en funksjon som tar inn to strenger og sjekker om de 3 første bokstavene i er lik i begge strengene
- Returner true eller false avhengig av om de er like eller ikke
- Det skal ikke spille noen rolle om det brukes små eller store bokstaver, input "Bord" og "BoRRemaskin" skal fremdeles gi "true"

Kont 2022 - TDT4109

OPPGAVE

(3a) Summere tall (5%)

Funksjonen **sum_pos_odd()** skal ta inn ei liste med heltall og returnere summen av de positive oddetallene i lista. Dvs., partall skal ikke tas med i summen, og negative tall skal heller ikke tas med.

Eksempel på bruk av funksjonen:

```
>>> sum_pos_odd([2,-3,5,-1,7,6,4])  
12
```

Svaret blir her 12 fordi de eneste positive oddetallene i lista er 5 og 7, og $5+7$ blir 12.

Skriv koden for funksjonen `sum_pos_odd()`

Øving 6 - Tekstbehandling

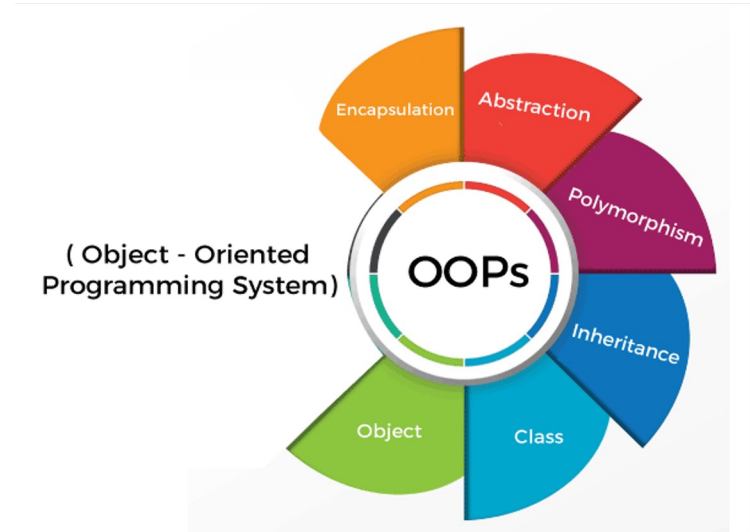
OPPGAVE

Lag en funksjon som tar inn en streng og en karakter som argumenter. Funksjonen skal splitte strengen med hensyn på denne karakteren, og returnere listen man får av denne splittingen.

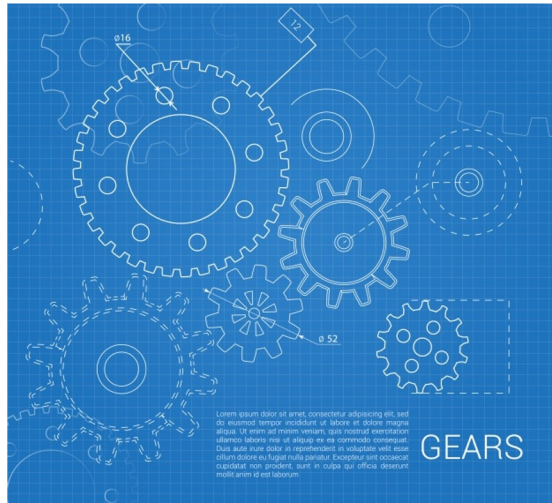
```
streng = "Hakuna Matata", karakter = 'a'  
output -> ['H', 'kun', ' M', 't', 't',  
          '']
```

(Hvis tid) Klasser og objekter

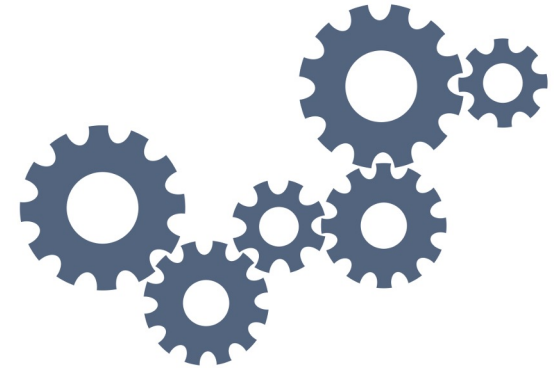
- Nå skal vi se litt nærmere på klasser og objektorientert programmering



Klasser og objekter



Klasse



Instansiering

Objekt

- Objekter er en del av et kjørende program som har
 - **Tilstand**
 - Data objektet “**husker**”
 - Lagres i variabler på samme måte som dere er kjent med fra ITGK. Disse variablene kalles **felt** i Java-terminologien
 - Tenk: Data som beskriver **dette ene** objektet
 - Evt: (se for deg data som beskriver spesifikt **deg** og ikke andre mennesker)
 - **Oppførsel**
 - Spørsmål du kan stille, tjenester du kan be objektet utføre
 - Oppførsel **endrer intern tilstand** over tid
- Fokuset i OOP er **oppførsel**, men tilstand og oppførsel utgjør en dualitet
 - Den ene kan sjelden eksistere uten den andre

- En klasse, er en **mal** for hvordan et sett av objekter skal se ut. Den kan beskrives som en "tredelt boks":
 - **Navn**
 - (identitet) som identifiserer klassen
 - **Felt**
 - (variabler/attributter/egenskap) som inneholder tilstanden til klassen
 - **Metoder**
 - (oppførsel/operasjon) som inneholder den dynamiske oppførselen til klassen
- Klassen **innkapsler** data (tilstand) og oppførsel
- Den tredelte boksen kalles gjerne et klassediagram

Gjennomgang av terminologi

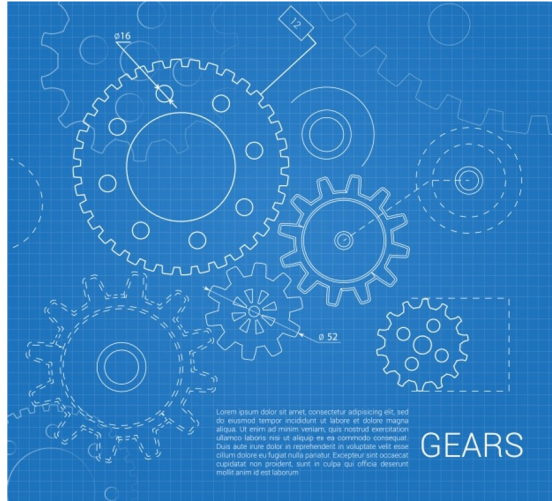
- **Deklarere**
 - Definere navnet og **datatypen** til en variabel
 - **int** number;
- **Initialisere**
 - Sette en verdi til en deklart variabel for første gang
 - **String** beskjed = "Hei, sveis!";
- **Metode**
 - Navn for **funksjoner** innenfor objektorientert.
 - Generelt brukes begrepet metoder når de er definert som en del av en klasse (ligger inne i klassedefinisjonen)
- **Instans**
 - Et konkret "eksempel" av noe
 - Et objekt er en instans av en klasse
 - Analogi: **Du** er en instans av "klassen" **Menneske**

Bestemme start-tilstand for objekt

- Er alle mennesker 100% like i begynnelsen?
 - Nei, man har f.eks. gener som definerer noen egenskaper ved seg
 - På samme måte må ikke alle nye objekter som instansieres av en klasse være “like” (i samme start-tilstand)
 - Vi kan definere parametere i **konstruktøren** til klassen som lar oss definere start-tilstanden

- **Konstruktøren** er en funksjon som kjøres når et objekt **instansieres**:
 - `Object obj = new Object(<arguments>);`
- Navnet på funksjonen **må** være det samme som klassenavnet.
- Av konvensjon er det normalt å definere konstruktørene øverst i klassedefinisjonen, under feltene, men det er ikke et krav.
- Det er fullt mulig å ha flere konstruktører i en klasse, men disse må ha unike **signaturer** for å separere de
 - Konstruktørene må ha forskjellig input-parametere
 - En annen måte å separere konstruktører fra hverandre er ved å bruke forskjellige **synlighetsmodifikatorer** som vi kommer til neste uke.
- Objekter **må** ikke ha definert konstruktør, men da vil alle objekter starte i samme tilstand*

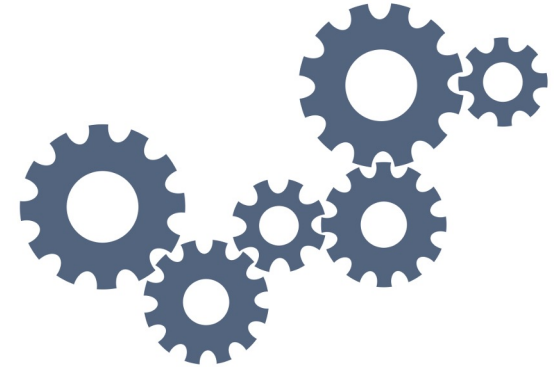
Klasser og objekter



Klasse

Konstruktøren kjøres
under **instansiering**

```
public Gears(...) {  
    ...  
}
```



Objekt

Getters og setters

- Et sett med standardmetoder i klasser som dere vil bli **godt kjent** med fremover
- **Setters**
 - Overskriver/modifiserer et internt felt (variabel) i klassen
 - Lar oss definere *regler* for hvordan feltene kan endres
- **Getters**
 - Returnerer et internt felt (variabel) i klassen
 - Hjelper oss med å *skjule implementasjonsdetaljer*
- Mulig å autogenerere disse med tillegg hvis man ønsker
- Vi kommer tilbake til hvorfor vi må ha getters og setters neste uke
 - Det er ingenting “spesielt” med disse metodene, kun *konvensjon*

Oppgave 8: CryptoCoin

OPPGAVE

- Lag klassen **CryptoCoin**
- En CryptoCoin har følgende attributter:
 - price (prisen for 1 enhet av coinen, målt i USD)
 - name (Navnet til coinen)
- Implementer get- og set-metoder for attributtene
- Lag en passende toString() - implementasjon

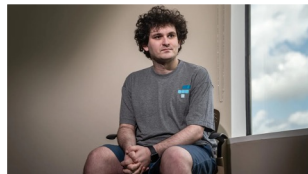


Oppgave 8: CryptoCoin

OPPGAVE

Lag de to følgende metodene:

- `toTheMoon()`
 - Denne metoden skal 10-doble prisen på en CryptoCoin
 - Metoden skal ikke returnere noe (void)
- `rugPull()`
 - Denne metoden skal sette prisen til 0, og returnerer verdien Coinen hadde rett før denne metoden ble kalt.



Neste uke

- Øvingsforelesning 1
 - Handler om øving 1
- Læringsassistentene begynner på jobb på mandag 22. januar
- Minner om påmelding av læringsassistentgrupper kl. 12 i dag!

Spørsmål og tilbakemeldinger kan sendes til
eirik.l.tanberg@ntnu.no