

## Oppmøtereistrering for smittesporing

- Scan QR-kode med egen mobil
- Logg på med FEIDE-bruker
- Registrer tidsrom og bordnummer
- <https://innsida.ntnu.no/checkin/room/38377>

Register your visit to  
**A4-112 Realfagbygget**  
QUBSHUGEN  
ROOM ID: 38377

0

Delaktene har sjekket inn på dette rommet under klokke 12:00



Manually check in to this room:  
[innsida.ntnu.no/checkin/room/38377](https://innsida.ntnu.no/checkin/room/38377)

1

# Repetisjon

2

2

## Byggeklossprinsippet i programmering



- Hvorfor lage flere klasser?
- Vi deler et komplekst problem opp i mindre deler og lager klasser for hver av dem
- En klasse er en logisk måte å **dele opp** et problem på
- En klasse beskriver **objekter**
- Vi kan **fokusere** på en klasse i gangen og lage denne mest mulig **uavhengig** av resten av totalsystemet
- Klasser kan testes og forandres uavhengig av hverandre
- Klasser kan **gjenbrukes** i andre sammenhenger
- I starten vil vi dele programsystemet i to klasser:
  - En klasse for å løse det aktuelle problemet
  - En klasse for å kommunisere med brukeren (**Bruker grensesnitt**)

3

3

## Byggeklossprinsippet i programmering



- Hvordan lage metodene i en klasse?
- Løser metoden min kun en oppgave?
- Bør jeg bruke metoder på flere nivåer?
- Bruker jeg anonyme konstanter i metoden som burde vært erstattet av parametre?
- Er navnet på metoden og parameterne så generelle som innholdet i metoden tilsier?
- Gjenbruk!!!!!!

4

4

# Tabeller av primitive datatyper, del 1

IDATT 1001 Programmering 1

Innstallasjon av IntelliJ  
Hva er en tabell  
Å kopiere tabeller  
Klassen Maned  
Tabeller og minneadministrasjon  
Sekvensielt søk i usortert tabell

5

## Agenda

Repetisjon, agenda og læringsutbytter

Installasjon av IntelliJ skal være gjort på forhånd

Demo: Lage HeiVerden i IntelliJ

Demo: Lage klassen Konto i IntelliJ

Lysark – tema: hvordan opprette og bruke en tabell  
Oppgave med tabeller

Lysark – tema: kopiering av en tabell  
Oppgave med kopiering av tabeller

Lysark – tema: system.arraycopy, klassen måned  
Oppgave med kopiering av tabeller

Lysark – tema: minneadministrasjon Oppgave med dyp og grunn  
kopiering Lysark – tema: søking i tabeller Oppgave med søking i tabeller

6

6

## Læringsutbytter for tabeller




- Emnenivå
  - har kunnskap om enkle prinsipper innen objektorientert programmering som innkapsling, modularisering og samhandlende objekter
  - kunne forklare hva som menes med en lagdelt arkitektur og hvorfor det er viktig i programvaredesign
- Detaljert
  - Avgjøre når vi har behov for strukturen tabell
  - Opprette en tabellstruktur i Java for primitive datatyper
  - Hente ut lengden til en tabell
  - Hente ut verdier fra en tabell
  - Fylle inn verdier i en tabell
  - Kopiere innholdet i en tabell over i en annen tabell
  - Programmere sekvensielle søk i en usortert tabell
  - Kunne gjøre rede for forskjellen mellom grunn og dyp kopiering av objekter
  - Vite hvilke unntak som er aktuelle i forbindelse med tabeller og hvordan håndtere disse

7

7

## Installasjon – IntelliJ IDEA



- IntelliJ IDEA is a cross-platform IDE that provides consistent experience on the Windows, macOS, and Linux operating systems.
- IntelliJ IDEA is available in the following editions:
  - **Community Edition** is free and open-source, licensed under Apache 2.0. It provides all the basic features for JVM and Android development.
  - **Ultimate Edition** is commercial, distributed with a 30-day trial period. It provides additional tools and features for web and enterprise development.
- Download IntelliJ IDEA
  - <https://www.jetbrains.com/idea/download/#section=windows>
  - <https://www.jetbrains.com/idea/download/#section=linux>
  - <https://www.jetbrains.com/idea/download/#section=mac>

8

8

## Demo

- Hello World
  - Prosjekt basert på helloworld template
- Klassen Account uten å skrive all kode selv
  - Code completion
  - Generer kode
- Hjelpetil med videoeksempel
  - [https://www.jetbrains.com/help/idea/2020.2/creating-and-running-your-first-java-application.html?utm\\_source=product&utm\\_medium=link&utm\\_campaign=IC&utm\\_content=2020.2](https://www.jetbrains.com/help/idea/2020.2/creating-and-running-your-first-java-application.html?utm_source=product&utm_medium=link&utm_campaign=IC&utm_content=2020.2)

9

9

```
import java.util.Objects;
import static javax.swing.JOptionPane.*;

class Account {
    long accountnr;
    String name;
    double saldo;

    public Account(long accountnr, String name, double saldo) {
        this.accountnr = accountnr;
        this.name = name;
        this.saldo = saldo;
    }

    public long getAccountnr() {
        return accountnr;
    }

    public String getName() {
        return name;
    }

    public double getSaldo() {
        return saldo;
    }

    public void setAccountnr(long accountnr) {
        this.accountnr = accountnr;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void setSaldo(double saldo) {
        this.saldo = saldo;
    }

    public void add(double amount) {
        saldo += amount;
    }

    public void withdraw(double amount) {
        saldo -= amount;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;
        Account account = (Account) o;
        return accountnr == account.accountnr &&
            Double.compare(account.saldo, saldo) == 0 &&
            Objects.equals(name, account.name);
    }

    @Override
    public String toString() {
        return "Account{" +
            "accountnr=" + accountnr +
            ", name=" + name + " " +
            ", saldo=" + saldo +
            '}';
    }
}
```

10

10

```

class TestAccount{

    public static void main(String[] args) {
        /* Oppretter et objekt av klassen Account. */
        Account olesAccount = new Account(123456676756L, "Ole Olsen", 2300.50);
        int counter = 1;
        while (counter != 0){ // avslutter dersom bruker skriver inn 0
            String readTransaksjon = showInputDialog("Velg transaksjon (avslutt, innskudd, uttak, saldo: ");
            char transaksjon = readTransaksjon.charAt(0);
            switch (transaksjon){
                case 'a': counter = 0; break;
                case 'i': String readDeposit = showInputDialog("Hvor mye skal du sette inn på konto: ");
                    int deposit = Integer.parseInt(readDeposit);
                    olesAccount.add(deposit);
                    if (olesAccount.getSaldo() < 0) showMessageDialog(null, "Etter innskudd har kontoen allikevel negativ saldo " + olesAccount.getSaldo());
                    else showMessageDialog(null, "Etter innskudd er saldoen lik " + olesAccount.getSaldo());
                    break;
                case 'u': String readWithdraw = showInputDialog("Hvor mye skal du ta ut fra konto: ");
                    int withdraw = -Integer.parseInt(readWithdraw);
                    olesAccount.withdraw(withdraw);
                    if (olesAccount.getSaldo() < 0) showMessageDialog(null, "Etter uttak har kontoen negativ saldo " + olesAccount.getSaldo());
                    else showMessageDialog(null, "Etter uttak er saldoen lik " + olesAccount.getSaldo());
                    break;
                case 's': showMessageDialog(null, "Din saldo er " + olesAccount.getSaldo()); break;
            } // end switch
        } // end while
    }
}

```

11

11

## Behov for datastrukturen tabell

- En *datastruktur* er en samling data lagret i primærminnet under ett navn.
- Et objekt er en datastruktur.
- En tabell er en datastruktur som kan brukes dersom alle dataene tilhører samme datatype.
- Eksempel: Nedbøren som faller hver dag i en måned.
- Hvordan deklare variablene?

```

int nedbør1;
int nedbør2;
int nedbør3;
int nedbør4;
.....
int nedbør31;

```

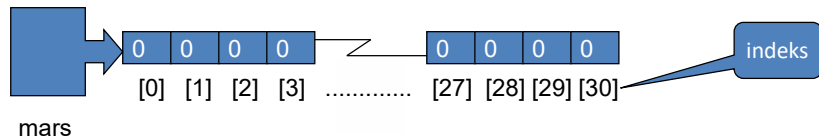
12

12

## Datastrukturen tabell

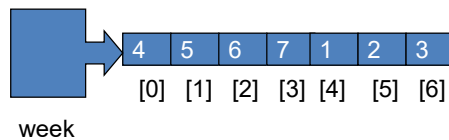
- Vi deklarerer en tabell:

```
- int[] mars = new int[31];
```



- Kan deklarer og initiere i én og samme prosess:

```
- int[] week = {4, 5, 6, 7, 1, 2, 3}; // lengden blir 7
```



13

13

## Å bruke en tabell

- Vi kan få tak i hvert enkelt element ved indeksring:

```
- mars[0] = 20;
```

```
- mars[15] = 30;
```

```
- int sum = mars[0] + mars[1] + mars[2];
```

- Lengde til tabellen (antall elementer)

```
- mars.length;
```

- Første element har alltid indeks 0, siste element indeks (length-1).

```
- int firstElement = mars[0];
```

```
- int lastElement = mars[mars.length-1];
```

14

14

## Å bruke en tabell

- Bruk av ugyldig indeks
  - `int number = mars[mars.length];`
  - kaster unntaket `ArrayIndexOutOfBoundsException`.
- Kontroller indeks **før** den brukes:
 

```
if (index >= 0 && index < mars.length) {
    number = mars[index];
}
```
- Summerer all nedbør i mars:
 

```
int sumMars = 0;
for (int i = 0; i < mars.length; i++) {
    sumMars += mars[i];
}
```

15

15

## Oppgaver

- a) Lag en tabell som skal ha plass til å lagre antall dager i hver måned. Gi verdier til tabellelementene ved initiering. Antall dager i februar settes lik 28.
  - b) Skriv en kodebit som spør bruker om det er et skuddår. Hvis det er det, skal antall dager i februar settes lik 29.
2. Lag en tabell av datatypen `char`. Den skal initieres med verdiene 'A', 'N', 'N', 'E'. Sett opp programkode som skriver ut disse verdiene i motsatt rekkefølge
3. Heltalls-tabellen `tab` initieres med verdiene 3, 8, -5, 5, 6, 0, 3, -2, 8, 9. Deretter utføres setningene under – hva inneholder tabellen nå?
 

```
tab[2] = tab[6] + 5;
int a = tab[8];
tab[7] = a + tab[0] * tab[0];
tab[4] = tab[4] + 1;
tab[5] = tab[3] + tab[9];
tab[3] = tab[2] * tab[0];
```

16

16



## Løsning på oppgaver

### Oppgave 1

```
import static javax.swing.JOptionPane.*;
class Oppg1{
    public static void main(String[] args) {
        int[] numberOfDays = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};
        int answer = showConfirmDialog(null, "Er det skuddår?", "År", YES_NO_OPTION);

        if (answer == YES_OPTION) numberOfDays[1] = 29;
        for (int i = 0; i < numberOfDays.length; i++)
            System.out.println(numberOfDays[i]);
    }
}
```

### Oppgave 2

```
char[] name = {'A', 'N', 'N', 'E'};
for (int i = name.length - 1; i >= 0; i--){ System.out.print(name[i]);
System.out.println(); // et linjeskift til slutt
```

### Oppgave 3

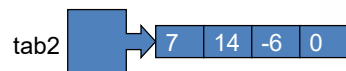
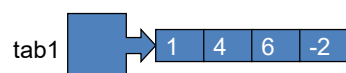
Tabellen har følgende innhold: 3, 8, 8, 3, 7, 14, 3, 17, 8, 9

17

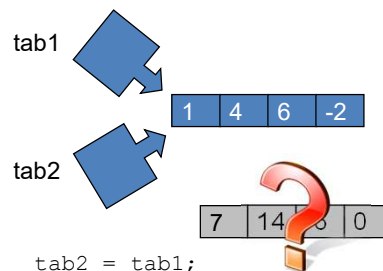
17

## Å kopiere en tabell

- Hva med å bruke tilordningstegnet (=)?



```
int[] tab1 = {1, 4, 6, -2};
int[] tab2 = {7, 14, -6, 0};
```



- Fungerer ikke ...vi får to referanser til samme tabell – og vi "mister" den andre tabellen.

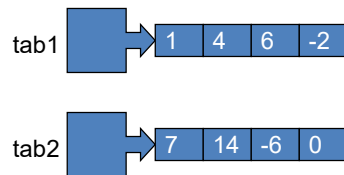
18

18

## Må kopiere en tabell element for element

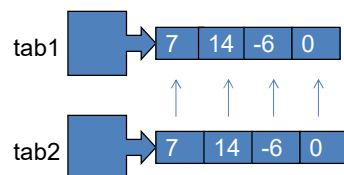
### Før kopiering

```
int[] tab1 = {1, 4, 6, -2};
int[] tab2 = {7, 14, -6, 0};
```



### Kopierer element for element

```
for (int i = 0; i <
    tab1.length; i++){
    tab1[i] = tab2[i];
}
```



19

## Oppgave

Tab1 og tab2 er gitt under. Sett opp en for-løkke som kopierer innholdet fra tab1 til tab2 i motsatt rekkefølge. Dvs at etter kopiering skal tab 2 inneholde verdiene -2, 6, 4, 1.

```
int [] tab1 ={1, 4, 6, -2};
int [] tab2 ={7, 14, -6, 0};
```

20

20

## Løsning

```
int [] tab1 = {1, 4, 6, -2};
int [] tab2 = {7, 14, -6, 0};

int tabLength = tab1.length; // lik for tabellene
for (int i = 0; i < tabLength; i++) {
    tab2[tabLength - 1 - i] = tab1[i];
}
```

21

21

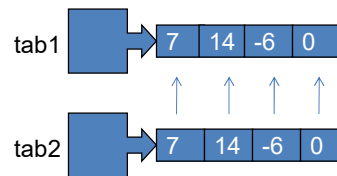
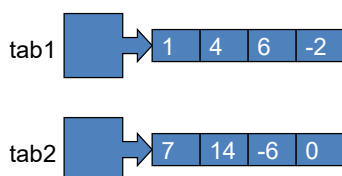
## Alternativ kopiering - Arrays.copyOf

### Før kopiering

```
int[] tab1 = {1, 4, 6, -2};
int[] tab2 = {7, 14, -6, 0};
```

### Kopierer ved hjelp av Arrays.copyOf

```
tab1 = Arrays.copyOf(tab2, tab2.length);
```



Hele tabellen kopieres

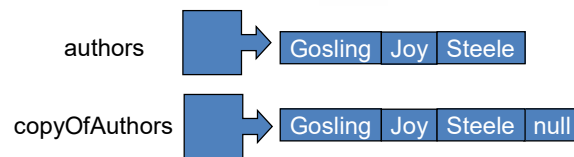
22

```
String[] copyofAuthors = Arrays.copyOf(authors,authors.length+2);
```

## Arrays.copyOf – kopiering av tabeller

- Vi kopierer hele tabellen og utvider den med 1
  - Arrays.copyOf(Objekt-from-table, int size)

```
String [] authors = {"Gosling", "Joy", "Steele",};
String[] copyofAuthors = Arrays.copyOf(authors,authors.length+1);
```



23

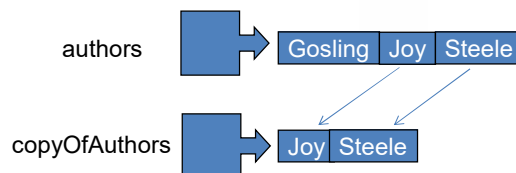
23

```
String[] copyofAuthors = Arrays.copyOfRange(authors,authors.length+2);
```

## Arrays.copyOf – kopiering av tabeller

- Arrays.copyOfRange(fromtable, fromIndex, toIndex)

```
String [] authors = {"Gosling", "Joy", "Steele",};
String[] copyofAuthors = Arrays.copyOfRange(authors,1,authors.length);
```



Kopieringsintervall: [fra, til>

24

24

## Klassen Maned

### Maned

```
-String monthName {readonly}
-int[] rain {readonly}

+Month(String monthName, int[]rain)
+String getMonthName()
+int findRain(int indeks)
+int findNumberOfDays()
+int findMax()
+int findNrOfDryDays()
+int finnAverage()

+int [] findMaks()
+int findNumberOfDaysMax()
+int findNumberOfDaysLess(int limit)
+int finnStandardDeviation()
```

Tabellen *rain* vil ha ulik lengde avhengig av hvilken måned det er snakk om. Januar: 31, april: 30, testmåned: 3...

25

25

## Oppgaver I TIMEN

I de tre første oppgavene skal du lage nye metoder i klassen Month. I den siste skal du legge inn feilkontroll i konstruktøren. Alt sammen skal prøves ut ved å utvide klientprogrammet.

1. Lag en metode som finner antall dager med nedbør lik maksimum.
2. Lag en metode som finner antall dager det har regnet færre millimeter enn et gitt verdi. Verdien skal være parameter.
3. Lag en metode som finner gjennomsnittlig avvik fra middelveien (standard avvik). Formelen må du eventuelt slå opp i ei statistikkbok eller søk på nett.

26

26

## Løsning oppgave 1

```
public int findNumberOfDaysMax() {  
    int counter = 0;  
    int max = findMax();  
    for(int i=0; i<rain.length; i++){  
        if (rain[i] == max) counter ++;  
    }  
    return counter;  
}
```

27

27

## Løsning oppgave 2

```
public int findNumberOfDaysLess(int limit){  
    int counter = 0;  
    for(int i=0; i<rain.length; i++){  
        if(rain[i]< limit) counter++;  
    }  
    return counter;  
}
```

28

28

## Løsning oppgave 3

```
public double findStdDeviation() {
    if (rain.length > 1) {
        int average = findAverage();
        long sumSquare = 0;
        for (int i = 0; i < rain.length; i++) {
            sumSquare += (average - rain[i]) * (average - rain[i]);
        }
        System.out.println(sumSquare);
        double radikand=(double)sumSquare/(double) (rain.length-1);
        return Math.sqrt(radikand); // Std.av beregnet etter formel
    } else {
        return 0.0; // For lite data til å beregne std.avvik
    }
}
```

29

29

## Minneadministrasjon

- Grunn kopiering / shallow copy
  - Setter referansen lik en annen uten å kopiere det referansen peker til
- Dyp kopiering / deep copy
  - Vi kopierer det referansen peker til

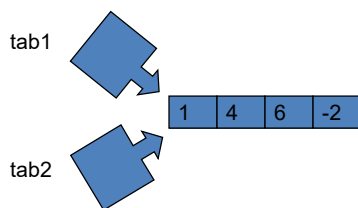
30

30

## Minneadministrasjon, kode-eksempel

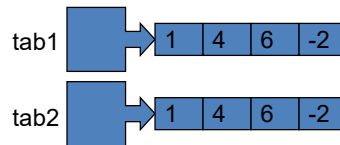
### Grunn kopiering

```
int tab1 = {1,4,6,-2};
int tab2 = tab1;
```



### Dyp kopiering

```
int tab1 = {1,4,6,-2};
int tab2 = new int[tab1.length];
for(int i=0; i<tab1.length; i++){
    Tab2[i] = tab1[i];
}
```



31

## Tabell som argument eller retur-verdi

- Normalsituasjon: Vi ønsker å **verne** om egne objektvariabler og beskytte disse fra omverdenen. Lag derfor dype kopier av:
  - tabeller som kommer inn som argument og som skal bli objektvariabler

```
public Rain(int [] rain){
    this.rain = new int[rain.length()];
    for(int i=0; i<rain.length();i++){
        this.rain[i] = rain[i];
    }
}
```

- tabeller som er objektvariabler og som sendes ut som returverdi fra en metode

```
public int[] getRain(){
    int[] tab = new int[rain.length()];
    for(int i=0; i<rain.length();i++){
        tab[i] = rain[i];
    }
    return tab; }
}
```

32

32



## Sekvensielt søk i usortert tabell

- Gjennomløper en tabell element for element, inntil man finner den verdien det søkes etter, eller man konstaterer at verdien ikke finnes.
- Må fortelle klienten om resultatet av søket – enten gi tilbake ønsket verdi eller en melding om verdi ikke funnet.
- Unngå uthopp flere steder i en metode – blir fort uoversiktlig
- boolean – nyttig datatype
- if (funnet)
  - gjør det som skal gjøres hvis verdien finnes
- else
  - gjør det som skal gjøres dersom verdien ikke finnes

33

33

## Kode-eksempler

### Uthopp inni i if

```
public int findDay(int value){
    for(int i=0; i<rain.length(); i++){
        if(rain[i] == value) {
            return i;
        }
    }
    return -1; // ikke funnet
}
```



Finner første forekomst av verdien

### Uthopp kun ett sted

```
public int findDay(int value){
    int result = -1;
    for(int i=0; i<rain.length(); i++){
        if(rain[i] == value) {
            result = i;
        }
    }
    return result;
}
```



Finner siste forekomst av verdien

34

34

## Mer kompliserte søkebetingelser – bruk boolean

```
public int findDay(int value){
    boolean found = false;
    int daynr = 0;
    while (daynr < rain.length && !found){
        if(rain[daynr] == value) {
            found = true;
        }else{
            daynr++;
        }
    }
    if (found){
        return daynr;
    }else{
        return -1;
    }
}
```

35

35

35

## Oppgave

- Klassen Arrays har metoder for å gjennomføre søk, lag en ny metode i klassen Maned som bruker denne.

```
public int findDay(int value){
    // fyll inn kode her..
}
```

- Hvilken søkealgoritme brukes for metoden og hvordan fungerer den?

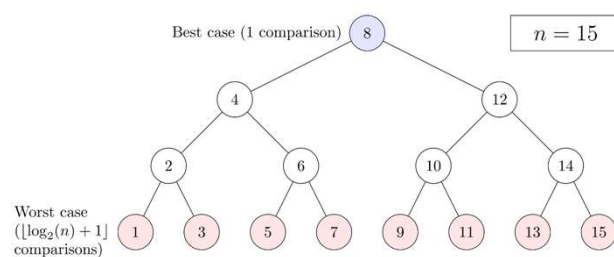
36

36

## Løsning

```
• public int findDay(int value) {  
    return Arrays.binarySearch(rain,  
    value);  
}
```

Bruker algoritmen binærsøk, forutsetter en sortert tabell



37

37