Rekursjon

Hva er rekursjon?

- En metode som kaller på seg selv
- Rekursive metoder trenger et *basistilfelle* og et *rekursjonssteg*
  - **Basistilfelle**
    - Når rekursjonen skal stoppe
    - Hva som skal skje da
  - **Rekursjonssteg**
    - Kall på seg selv
    - Hva som skal endres

```
private static int teller(int t) {

    System.out.println("Tallet er: " + t);
    return teller(t + 1);
}
```

Hva er rekursjon?

- En metode som kaller på seg selv
- Rekursive metoder trenger et *basistilfelle* og et *rekursjonssteg*
  - **Basistilfelle**
    - Når rekursjonen skal stoppe
    - Hva som skal skje da
  - **Rekursjonssteg**
    - Kall på seg selv
    - Hva som skal endres

```
private static int teller(int t) {

    if (t == 1000) return 0;

    System.out.println("Tallet er: " + t);
    return teller(t + 1);
}
```

Call Stack

- I java legges metodekall i en *stack*
- Metoden som kalles på nå ligger øverst i stacken
- Slik “husker” Java hvor vi er i koden

Et lite eksempel

```java
public static void main(String[] args) {
    metodeA();
}

public static void metodeA() {
    System.out.println("Hei fra metode A!")
    metodeB();
    System.out.println("Hade fra metode A!")
}

public static void metodeB() {
    System.out.println("Hei fra metode B!");
}
```

main()

Et lite eksempel



```java
public static void main(String[] args) {
    metodeA();
}

public static void metodeA() {
    System.out.println("Hei fra metode A!")
    metodeB();
    System.out.println("Hade fra metode A!")
}

public static void metodeB() {
    System.out.println("Hei fra metode B!");
}
```

main()

Et lite eksempel

```java
public static void main(String[] args) {
    metodeA();
}

public static void metodeA() {
    System.out.println("Hei fra metode A!")
    metodeB();
    System.out.println("Hade fra metode A!")
}

public static void metodeB() {
    System.out.println("Hei fra metode B!");
}
```



metodeA()

main()

Hei fra metode A!

Et lite eksempel

```java
public static void main(String[] args) {
    metodeA();
}

public static void metodeA() {
    System.out.println("Hei fra metode A!")
    metodeB();
    System.out.println("Hade fra metode A!")
}

public static void metodeB() {
    System.out.println("Hei fra metode B!");
}
```



metodeA()

main()

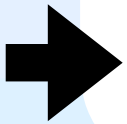
Hei fra metode A!

Et lite eksempel

```
public static void main(String[] args) {
    metodeA();
}

public static void metodeA() {
    System.out.println("Hei fra metode A!")
    metodeB();
    System.out.println("Hade fra metode A!")
}

public static void metodeB() {
    System.out.println("Hei fra metode B!");
}
```



metodeB()

metodeA()

main()

Hei fra metode A!

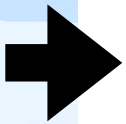
Hei fra metode B!

Et lite eksempel

```java
public static void main(String[] args) {
    metodeA();
}

public static void metodeA() {
    System.out.println("Hei fra metode A!")
    metodeB();
    System.out.println("Hade fra metode A!")
}

public static void metodeB() {
    System.out.println("Hei fra metode B!");
}
```



metodeA()

main()

Hei fra metode A!

Hei fra metode B!

Hade fra metode A!

Et lite eksempel



```java
public static void main(String[] args) {
    metodeA();
}

public static void metodeA() {
    System.out.println("Hei fra metode A!")
    metodeB();
    System.out.println("Hade fra metode A!")
}

public static void metodeB() {
    System.out.println("Hei fra metode B!");
}
```

main()

Hei fra metode A!

Hei fra metode B!

Hade fra metode A!

Hvordan ser dette ut i vår rekursive teller?

```java
public static void main(String[] args) {
    teller(0);
}

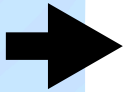
private static int teller(int t) {

    if (t == 10) return 0;

    System.out.println("Tallet er: " + t);
    return teller(t + 1);
}
```

main()

Hvordan ser dette ut i vår rekursive teller?



```java
public static void main(String[] args) {
    teller(0);
}

private static int teller(int t) {

    if (t == 10) return 0;

    System.out.println("Tallet er: " + t);
    return teller(t + 1);
}
```

teller(0)
main()

Hvordan ser dette ut i vår rekursive teller?

```
public static void main(String[] args) {
    teller(0);
}

private static int teller(int t) {

    if (t == 10) return 0;

    System.out.println("Tallet er: " + t);
    return teller(t + 1);
}
```

teller(0)
main()

Hvordan ser dette ut i vår rekursive teller?

```
public static void main(String[] args) {
    teller(0);
}

private static int teller(int t) {

    if (t == 10) return 0;

    System.out.println("Tallet er: " + t);
    return teller(t + 1);
}
```

teller(0)
main()

Tallet er: 0

Hvordan ser dette ut i vår rekursive teller?

```
public static void main(String[] args) {
    teller(0);
}

private static int teller(int t) {

    if (t == 10) return 0;

    System.out.println("Tallet er: " + t);
    return teller(t + 1);
}
```

teller(1)
teller(0)
main()

Tallet er: 0

Hvordan ser dette ut i vår rekursive teller?

```
public static void main(String[] args) {
    teller(0);
}

private static int teller(int t) {

    if (t == 10) return 0;

    System.out.println("Tallet er: " + t);
    return teller(t + 1);
}
```

teller(1)

teller(0)

main()

Tallet er: 0

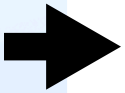
Hvordan ser dette ut i vår rekursive teller?

```
public static void main(String[] args) {
    teller(0);
}

private static int teller(int t) {

    if (t == 10) return 0;

    System.out.println("Tallet er: " + t);
    return teller(t + 1);
}
```



teller(1)

teller(0)

main()

Tallet er: 0

Tallet er: 1

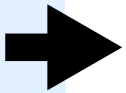
Hvordan ser dette ut i vår rekursive teller?

```
public static void main(String[] args) {
    teller(0);
}

private static int teller(int t) {

    if (t == 10) return 0;

    System.out.println("Tallet er: " + t);
    return teller(t + 1);
}
```



teller(2)
teller(1)
teller(0)
main()

Tallet er: 0
Tallet er: 1

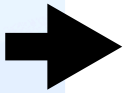
Hvordan ser dette ut i vår rekursive teller?

```
public static void main(String[] args) {
    teller(0);
}

private static int teller(int t) {

    if (t == 10) return 0;

    System.out.println("Tallet er: " + t);
    return teller(t + 1);
}
```



teller(2)
teller(1)
teller(0)
main()

Tallet er: 0
Tallet er: 1
Tallet er: 2

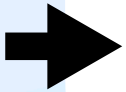
Hvordan ser dette ut i vår rekursive teller?

```
public static void main(String[] args) {
    teller(0);
}

private static int teller(int t) {

    if (t == 10) return 0;

    System.out.println("Tallet er: " + t);
    return teller(t + 1);
}
```



teller(3)
teller(2)
teller(1)
teller(0)
main()

Tallet er: 0
Tallet er: 1
Tallet er: 2

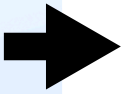
Hvordan ser dette ut i vår rekursive teller?

```java
public static void main(String[] args) {
    teller(0);
}

private static int teller(int t) {

    if (t == 10) return 0;

    System.out.println("Tallet er: " + t);
    return teller(t + 1);
}
```



teller(3)
teller(2)
teller(1)
teller(0)
main()

Tallet er: 0
Tallet er: 1
Tallet er: 2
Tallet er: 3

Fibonaccitall

- Lag et program som regner ut det n-te fibonaccitallet
  - Hva må basissteget være?
  - Hva er rekursjonssteget?
  - Begynn eventuelt med en ikke-rekursiv løsning 😉

Lenkelister

og rekursive metoder

- Lag en enkeltlenket liste-klasse med kun en start-peker
- Implementer følgende metoder rekursivt
  - slettSiste() - *fjerner siste node fra listen*
  - get(int index) - *henter ut verdi på gitt index*
  - flipListe() - *reverserer listen*