Mentimeter

Instructions

Mentimeter

Hvordan har innlevering 0 gått?



1
11

Mentimeter

Hva håper dere å få ut av gruppetimene?

Repetisjon/oppgaveløsing

Løse oppgaver

Oppgavehjelp+supplement

Oblig hjelp

oppgavehjelp, repitisjon, supplerment

oppgave hjelp

Jeg vil se Live koding og liker at forklaringene går sakte spesielt når koden blir lengre og lengre.

Jobbe med programmering

9

Mentimeter

Hva håper dere å få ut av gruppetimene?

Livekoding

Kjapp gjennomgang og mye tid til oppgaver

9

Mentimeter

Agenda for i dag

- Lesing fra fil
- Skriving til fil
- Todimensjonale arrays
- Metoder i java
- char vs String
- String-operasjoner
- Hvordan å bruke static

Mentimeter

Hvordan vil dere helst ha gruppetimene?

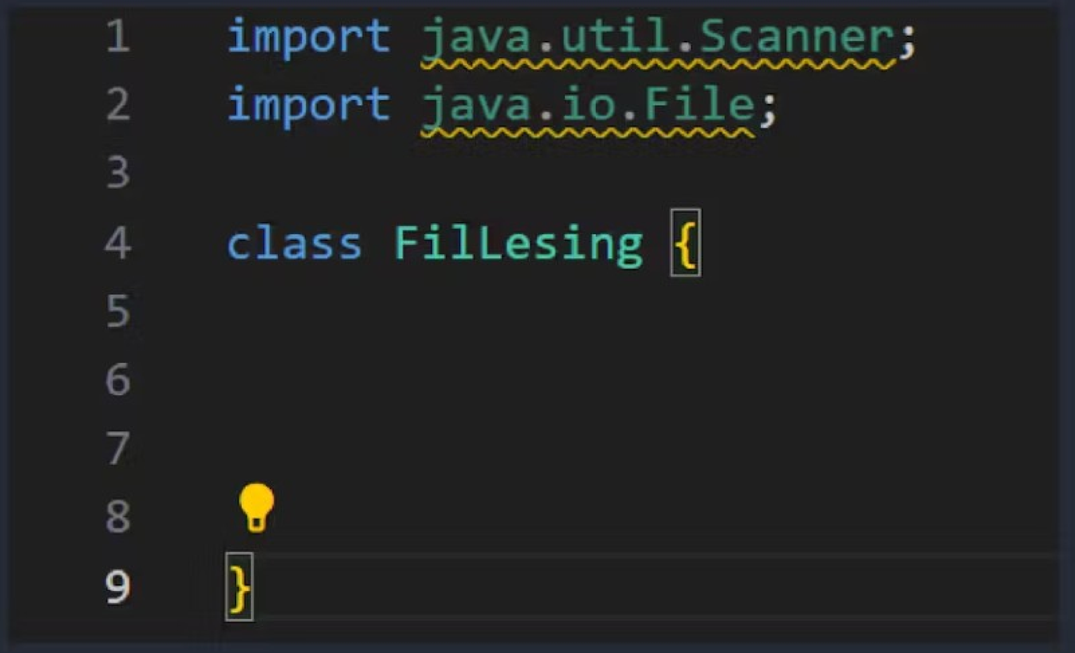


7

Mentimeter

Lesing fra fil ved hjelp av variabel

1. Vi starter med å lage skjelettet av klassen vår.



Mentimeter

Lesing fra fil ved hjelp av variabel

1. Vi starter med å lage skjelettet av klassen vår.
2. Vi trenger en variabel for filnavnet og en variabel som åpner opp filen.

```java
import java.util.Scanner;
import java.io.File;

class FilLesing {


    Run | Debug
    public static void main(String[] args) {
        String filnavn = "data.txt";
        File fil = new File(filnavn);



    }

}
```

Mentimeter

Lesing fra fil ved hjelp av variabel

1. Vi starter med å lage skjelettet av klassen vår.
2. Vi trenger en variabel for filnavnet og en variabel som åpner opp filen.
3. Lager en scanner som leser fra filen vår.
4. Kjører en loop som ikke stopper før vi har lest alle linjene i filen.

```java
import java.util.Scanner;
import java.io.File;

class FilLesing {


    Run | Debug
    public static void main(String[] args) {
        String filnavn = "data.txt";
        File fil = new File(filnavn);


        try {
            Scanner filleser = new Scanner(fil);

            while (filleser.hasNextLine()) {
                String linje = filleser.nextLine();

                System.out.println(linje);
            }

            filleser.close();
        }
```

Mentimeter

Lesing fra fil ved hjelp av variabel

1. Vi starter med å lage skjelettet av klassen vår.
2. Vi trenger en variabel for filnavnet og en variabel som åpner opp filen.
3. Lager en scanner som leser fra filen vår.
4. Kjører en loop som ikke stopper før vi har lest alle linjene i filen.
5. Alt dette ligger i en try-catch som sørger for at programmet ikke krasjer hvis filen ikke blir funnet.

```
11
12      try {
13          Scanner filleser = new Scanner(fil);
14
15          while (filleser.hasNextLine()) {
16              String linje = filleser.nextLine();
17
18              System.out.println(linje);
19          }
20
21          filleser.close();
22      }
23
24      catch (Exception e) {
25          System.out.println("Fant ingen fil med navn " + filnavn);
26          System.err.println(e);
27      }
```

Mentimeter

Lesing fra fil ved hjelp av terminalen

Mye likt, men må endre litt på hva som skjer med skanner.

- Vi trenger fortsatt en variabel for filnavnet og en variabel som åpner opp filen, men først må vi lage en scanner som kan lese fra terminalen.

```
4   class FilLesing {
5
6
        Run | Debug
7       public static void main(String[] args) {
8
9           Scanner lesTerminal = new Scanner(System.in);
10          String filnavn = lesTerminal.nextLine();
11          File fil = new File(filnavn);
12
13
14          try {
15              Scanner filleser = new Scanner(fil);
16
17              while (filleser.hasNextLine()) {
18                  String linje = filleser.nextLine();
19
20                  System.out.println(linje);
21              }
22
23              filleser.close();
24          }
```

Mentimeter

Lesing fra fil ved hjelp av terminalen

Mye likt, men må endre litt på hva som skjer med skanner.

- Vi trenger fortsatt en variabel for filnavnet og en variabel som åpner opp filen, men først må vi lage en scanner som kan lese fra terminalen.
- Er det en måte å løse dette med kun én scanner?

```
4   class FilLesing {
5
6
        Run | Debug
7       public static void main(String[] args) {
8
9           Scanner lesTerminal = new Scanner(System.in);
10          String filnavn = lesTerminal.nextLine();
11          File fil = new File(filnavn);
12
13
14          try {
15              Scanner filleser = new Scanner(fil);
16
17              while (filleser.hasNextLine()) {
18                  String linje = filleser.nextLine();
19
20                  System.out.println(linje);
21              }
22
23              filleser.close();
24          }
```

Mentimeter

Lesing fra fil som argument ved kjøring

Her trenger vi å se på args-variabelen

- Bruker args som filnavnet vårt.

```
Run | Debug
7   public static void main(String[] args) {
8
9       File fil = new File(args[0]);
10
11
12      try {
13          Scanner filleser = new Scanner(fil);
14
15          while (filleser.hasNextLine()) {
16              String linje = filleser.nextLine();
17
18              System.out.println(linje);
19          }
20
21          filleser.close();
22      }
23
24      catch (Exception e) {
25          System.out.println("Fant ingen fil med navn " + args[0]);
26          System.err.println(e);
27      }
```

Mentimeter

Skrive til en fil

- Mye likt som når vi leser filer, men nå bruker vi en PrintWriter.
- OBS! Hvis det står noe i filen fra før så vil det bli overskrevet.

```java
import java.io.File;
import java.io.PrintWriter;

class Filskriving {


    Run | Debug
    public static void main(String[] args) {
        File fil = new File(pathname:"data.txt");

        try {
            PrintWriter skriver = new PrintWriter(fil);

            skriver.println(x:"hei :)");

            skriver.close();
        }

        catch (Exception e) {
            System.out.println(x:"Fant ikke fil med angitt filnavn");
        }
    }
}
```

Mentimeter

Todimensjonale arrays

- Det lages ganske likt som vanlige arrays.

```
1   class Todimensjonale {
2
        Run | Debug
3       public static void main(String[] args) {
4
5           int[][] toDim = new int[5][5];
6
7       }
8
9
10  }
```

Mentimeter

Todimensjonale arrays

- Det lages ganske likt som vanlige arrays.
- Ofte vet vi hvor stor den skal være allerede før vi lager den.

```
1   class Todimensjonale {
2
        Run | Debug
3       public static void main(String[] args) {
4
5           int rad = 5;
6           int kol = 5;
7
8           int[][] toDim = new int[rad][kol];
9
10      }
11
12
13  }
```

Mentimeter

Todimensjonale arrays

- Det lages ganske likt som vanlige arrays.
- Ofte vet vi hvor stor den skal være allerede før vi lager den.
- Når vi skal traversere den må vi bruke to for-loops. En for rad og en for kolonne.

```java
class Todimensjonale {

    Run | Debug
    public static void main(String[] args) {

        int rad = 5;
        int kol = 5;

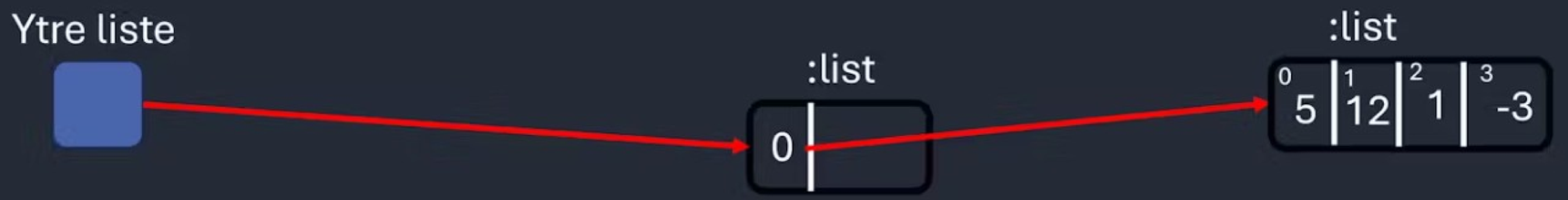
        int[][] toDim = new int[rad][kol];


        for (int i = 0; i < rad; i++) {
            for (int j = 0; j < kol; j++) {
                int tall = toDim[i][j];
            }
        }
    }


}
```

Mentimeter

Todimensjonale arrays



Mentimeter

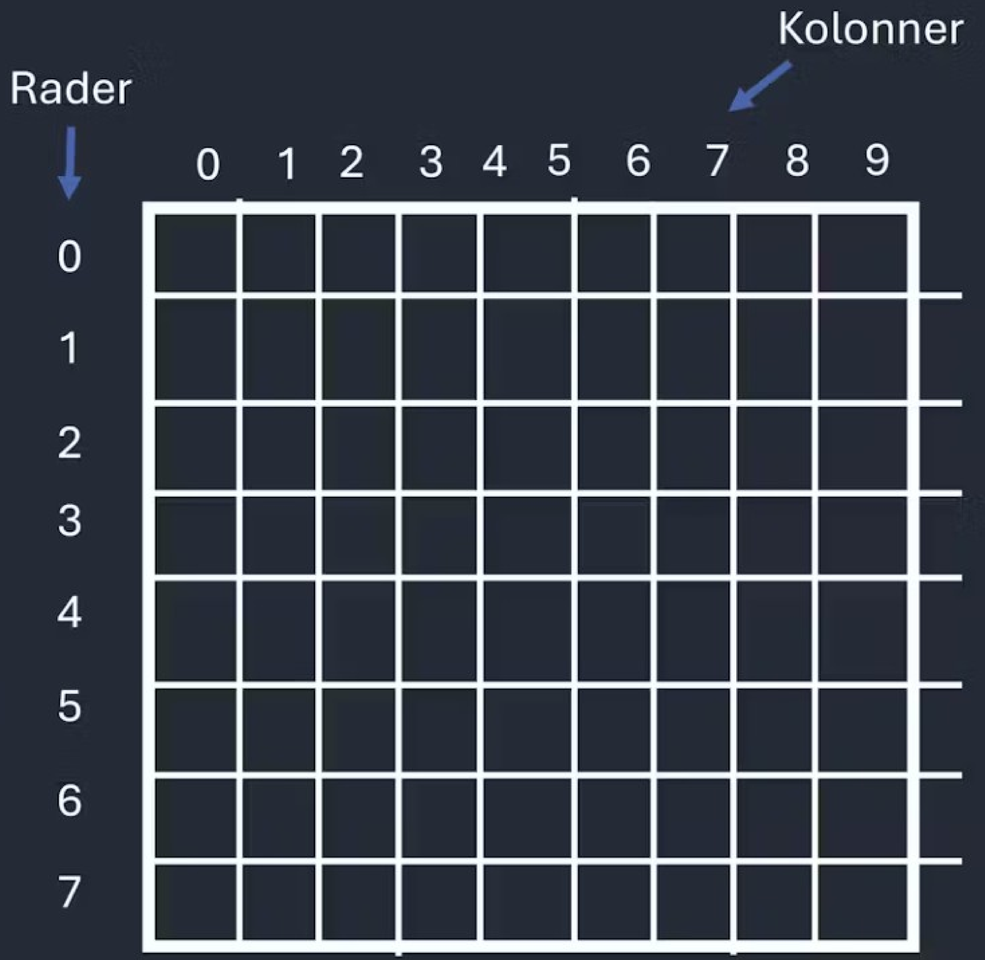
Todimensjonale arrays



Mentimeter

Todimensjonale arrays





Mentimeter

Metoder i java

- Metoder i java er klasse-spesifikke og lages litt annerledes enn i python.
- Vi må deklarere hva som returneres, og hvis ikke noe skal returneres så skriver vi void.
- Vi kan bestemme hvor tilgjengelig metoden er, enten public, private, protected eller unngå å deklarere noe.
- La oss livekode litt hvordan å lage objekter av denne klassen og hvordan å lage konstruktører ←(__init__ fra python).

```java
class Metoder {

    int tall = 0;

    void siHei() {
        System.out.println(x:"Hei!");
    }

    private void endreTall(int tall) {
        this.tall = tall;
    }

    public void siHade() {
        System.out.println(x:"Hade :(");
    }


    int hentTall() {
        return tall;
    }
}
```

Mentimeter

char vs String

- En char i java er et unikt tegn, mens en String er en kjede med tegn som ikke kan endres.
- Objekter som ikke kan endres etter opprettelse kaller vi *immutable*.

```
1   class CharString {
2
        Run | Debug
3       public static void main(String[] args) {
4           String tekst = "Hei på deg";
5           char karakter = 'H';
6       }
7   }
```

Mentimeter

String-operasjoner

- Vi kan kombinere to tekst-strenger ved ‘+’-operatoren.

```
1   class CharString {
2
        Run | Debug
3       public static void main(String[] args) {
4           char karakter = 'H';
5           String tekst = "Hei på deg";
6           String tekst2 = "Hade bra";
7
8           String nyTekst = tekst + tekst2;
9       }
10  }
```

Mentimeter

String-operasjoner

- Vi kan kombinere to tekst-strenger ved '+'-operatoren.
- Vi kan finne lengden til en streng med string.length().

```
1   class CharString {
2
        Run | Debug
3       public static void main(String[] args) {
4           char karakter = 'H';
5           String tekst = "Hei på deg";
6           String tekst2 = "Hade bra";
7
8           String nyTekst = tekst + tekst2;
9           int lengde = tekst.length(); // <-- gir 10
10      }
11  }
```

Mentimeter

String-operasjoner

- Vi kan kombinere to tekst-strenger ved '+'-operatoren.
- Vi kan finne lengden til en streng med string.length().
- String.trim() brukes for å fjerne '\n'-tegnet. Dette er ofte nyttig når vi leser fra filer for å fjerne ekstra linjeskift.
- String.split('x') splitter teksten opp i et array og fjerner tegnet vi ikke ønsker. Ofte splitter vi ved mellomrom.

```java
class CharString {

    Run | Debug
    public static void main(String[] args) {
        char karakter = 'H';
        String tekst = "Hei på deg";
        String tekst2 = "Hade bra";

        String nyTekst = tekst + tekst2;
        int lengde = tekst.length(); // <-- gir 10

        tekst = "Hei på deg\n";
        tekst = tekst.trim();
        System.out.println(tekst); // <-- "Hei på deg"

        String[] ord = tekst.split(regex:" ");
        System.out.println(ord[0]+ord[1]+ord[2]); // <-- Heipådeg

    }
}
```

Mentimeter

String-operasjoner

- Vi kan kombinere to tekst-strenger ved '+'-operatoren.
- Vi kan finne lengden til en streng med string.length().
- String.trim() brukes for å fjerne '\n'-tegnet. Dette er ofte nyttig når vi leser fra filer for å fjerne ekstra linjeskift.
- String.split('x') splitter teksten opp i et array og fjerner tegnet vi ikke ønsker. Ofte splitter vi ved mellomrom.
- Vi kan hente ut en karakter på en spesifikk lokasjon ved å bruke string.charAt(i)

```java
class CharString {

    Run | Debug
    public static void main(String[] args) {
        char karakter = 'H';
        String tekst = "Hei på deg";
        String tekst2 = "Hade bra";

        String nyTekst = tekst + tekst2;
        int lengde = tekst.length(); // <-- gir 10

        tekst = "Hei på deg\n";
        tekst = tekst.trim();
        System.out.println(tekst); // <-- "Hei på deg"

        String[] ord = tekst.split(regex:" ");
        System.out.println(ord[0]+ord[1]+ord[2]); // <-- Heipådeg

        System.out.println(tekst.charAt(index:4)); // <-- p
    }
}
```

Mentimeter

String-operasjoner

- Vi kan kombinere to tekst-strenger ved '+'-operatoren.
- Vi kan finne lengden til en streng med string.length().
- String.trim() brukes for å fjerne '\n'-tegnet. Dette er ofte nyttig når vi leser fra filer for å fjerne ekstra linjeskift.
- String.split('x') splitter teksten opp i et array og fjerner tegnet vi ikke ønsker. Ofte splitter vi ved mellomrom.
- Vi kan hente ut en karakter på en spesifikk lokasjon ved å bruke string.charAt(i).
- Sammenligning av to strenger gjør vi ved å bruke string1.equals(string2), ikke ved bruk av ==

```
Run | Debug
3   public static void main(String[] args) {
4       char karakter = 'H';
5       String tekst = "Hei på deg";
6       String tekst2 = "Hade bra";
7
8       String nyTekst = tekst + tekst2;
9       int lengde = tekst.length(); // <-- gir 10
10
11      tekst = "Hei på deg\n";
12      tekst = tekst.trim();
13      System.out.println(tekst); // <-- "Hei på deg"
14
15      String[] ord = tekst.split(regex:" ");
16      System.out.println(ord[0]+ord[1]+ord[2]); // <-- Heipådeg
17
18      System.out.println(tekst.charAt(index:4)); // <-- p
19
20
21      System.out.println(tekst.equals(tekst2));
22  }
23  }
```

Mentimeter

Static – Hva, hvor og når

- Static variabler/metoder deles av hele klassen. De er altså ikke unike for hvert objekt, men deles av alle objekter i klassen.
- Brukes ofte for å lage verdier som skal være unike for hvert objekt i klassen.

```
1   class Medlem {
2       static int nesteID = 0;
3       int id;
4
5       public Medlem() {
6           id = nesteID;
7           nesteID++;
8       }
9   }
```

Mentimeter

0 questions
0 upvotes