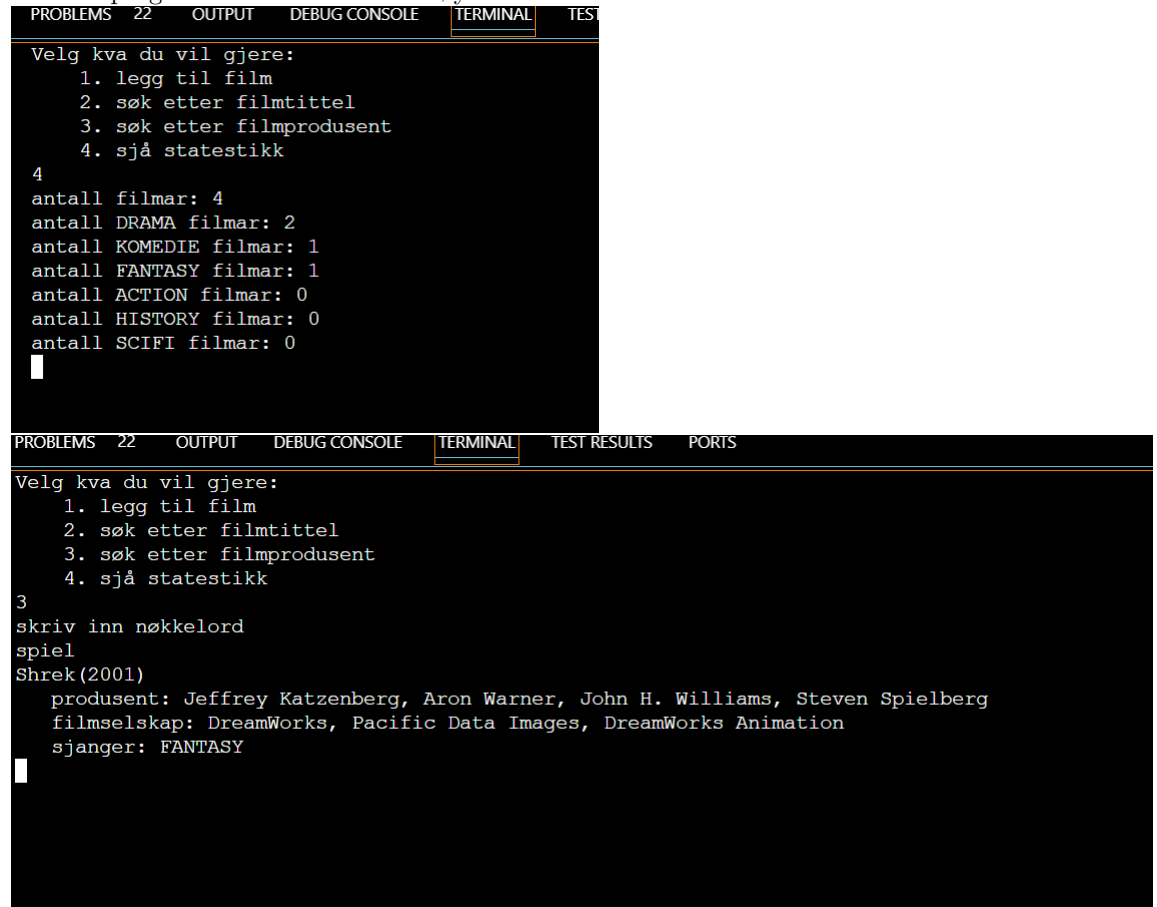


Kven er med på Gruppa?

Holly Storøy

1 Oppgave 1

her er sånn programmet ser ut når ein køyrer det.



```
PROBLEMS 22 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS
Velg kva du vil gjere:
  1. legg til film
  2. søk etter filmtittel
  3. søk etter filmprodusent
  4. sjå statestikk
4
antall filmar: 4
antall DRAMA filmar: 2
antall KOMEDIE filmar: 1
antall FANTASY filmar: 1
antall ACTION filmar: 0
antall HISTORY filmar: 0
antall SCIFI filmar: 0
█

PROBLEMS 22 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS
Velg kva du vil gjere:
  1. legg til film
  2. søk etter filmtittel
  3. søk etter filmprodusent
  4. sjå statestikk
3
skriv inn nøkkelord
spiel
Shrek(2001)
  produsent: Jeffrey Katzenberg, Aron Warner, John H. Williams, Steven Spielberg
  filmselskap: DreamWorks, Pacific Data Images, DreamWorks Animation
  sjanger: FANTASY
█
```

```
PROBLEMS 22  OUTPUT  DEBUG CONSOLE  TERMINAL  TEST RESULTS  PORTS

Velg kva du vil gjere:
  1. legg til film
  2. søk etter filmtittel
  3. søk etter filmprodusent
  4. sjå statestikk

2
skriv inn nøkkelord
The
The Room(2003)
  produsent: Tommy Wiseau
  filmselskap: Wiseau-Films
  sjanger: DRAMA
```

```
Velg kva du vil gjere:
  1. legg til film
  2. søk etter filmtittel
  3. søk etter filmprodusent
  4. sjå statestikk

2
skriv inn nøkkelord

Mannen som elsket Yngve(2008)
  produsent: Yngve Sæther
  filmselskap: Motlys, Sandrew Metronome
  sjanger: DRAMA
Shrek(2001)
  produsent: Jeffrey Katzenberg, Aron Warner, John H. Williams, Steven Spielberg
  filmselskap: DreamWorks, Pacific Data Images, DreamWorks Animation
  sjanger: FANTASY
The Room(2003)
  produsent: Tommy Wiseau
  filmselskap: Wiseau-Films
  sjanger: DRAMA
Click(2006)
  produsent: Adam Sandler, Jack Giarraputo, Neal H. Moritz, Steve Koren, Mark O'Keefe
  filmselskap: Columbia Pictures, Revolution Studios, Happy Madison Productions, Original Film, Sony
  sjanger: KOMEDIE
```

```
PROBLEMS 22 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS

Velg kva du vil gjere:
  1. legg til film
  2. søk etter filmtittel
  3. søk etter filmprodusent
  4. sjå statestikk

1
Skriv inn filmTittel:
Film
Skriv inn produsent:
Ein produsent
Skriv inn lanseringsår:
2025
Skriv inn sjanger:
SCI
Skriv inn filmselskap:
NRK
Film lagt til
█
```

```
Velg kva du vil gjere:
  1. legg til film
  2. søk etter filmtittel
  3. søk etter filmprodusent
  4. sjå statestikk

2
skriv inn nøkkelord

Mannen som elsket Yngve(2008)
  produsent: Yngve Sæther
  filmselskap: Motlys, Sandrew Metronome
  sjanger: DRAMA
Shrek(2001)
  produsent: Jeffrey Katzenberg, Aron Warner, John H. Williams, Steven Spielberg
  filmselskap: DreamWorks, Pacific Data Images, DreamWorks Animation
  sjanger: FANTASY
The Room(2003)
  produsent: Tommy Wiseau
  filmselskap: Wiseau-Films
  sjanger: DRAMA
Click(2006)
  produsent: Adam Sandler, Jack Giarraputo, Neal H. Moritz, Steve Koren, Mark O'Keefe
  filmselskap: Columbia Pictures, Revolution Studios, Happy Madison Productions, Original Film, Sony
  sjanger: KOMEDIE
Film(2025)
  produsent: Ein produsent
  filmselskap: NRK
  sjanger: SCIFI
█
```

```
PROBLEMS 22 OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS

Velg kva du vil gjere:
  1. legg til film
  2. søk etter filmtittel
  3. søk etter filmprodusent
  4. sjå statestikk
4
antall filmar: 5
antall DRAMA filmar: 2
antall KOMEDIE filmar: 1
antall FANTASY filmar: 1
antall ACTION filmar: 0
antall HISTORY filmar: 0
antall SCIFI filmar: 1
█
```

2 Oppgåve 2

Her er den nodebaserte implementasjonen av filmarkiv

```
1 package no.hvl.dat102.filmarkiv.impl;
2
3 import no.hvl.dat102.filmarkiv.adt.FilmarkivADT;
4
5 public class Filmarkiv2 implements FilmarkivADT {
6     public class LinearNode<T> {
7         public T data = null;
8         public LinearNode<T> neste = null;
9     }
10
11     private int antall = 0;
12     private LinearNode<Film> head = null;
13
14     @Override
15     public Film finnFilm(int nr) {
16         LinearNode<Film> current = head;
17         do {
18             if (current.data.getFilmNr() == nr){
19                 return current.data;
20             }
21             current = current.neste;
22         } while (current.neste != null);
23         return null;
24     }
25
26     @Override
```

```

27     public void leggTilFilm(Film nyFilm) {
28         LinearNode<Film> nyHead = new LinearNode<>();
29         nyHead.neste = head;
30         nyHead.data = nyFilm;
31         head = nyHead;
32         antall++;
33     }
34
35     @Override
36     public boolean slettFilm(int filmnr) {
37         LinearNode<Film> current = head;
38         if (head.data.getFilmNr() == filmnr){
39             head = head.neste;
40             antall--;
41             return true;
42         }
43         while (current != null) {
44             if (current.neste.data.getFilmNr() == filmnr){
45                 current.neste = current.neste.neste;
46                 antall--;
47                 return true;
48             }
49             current = current.neste;
50         }
51         return false;
52     }
53
54     @Override
55     public Film[] soekTittel(String delstreng) {
56         LinearNode<Film> current = head;
57         Filmarkiv2 ret = new Filmarkiv2();
58         while (current != null) {
59             if (current.data.getTittel().toLowerCase().contains(
60 delstreng.toLowerCase())){
61                 ret.leggTilFilm(current.data);
62             }
63             current = current.neste;
64         }
65         return ret.toArray();
66     }
67
68     @Override
69     public Film[] soekProdusent(String delstreng) {
70         LinearNode<Film> current = head;
71         Filmarkiv2 ret = new Filmarkiv2();
72         while (current != null) {
73             if (current.data.getProdusent().toLowerCase().contains(
74 delstreng.toLowerCase())){
75                 ret.leggTilFilm(current.data);
76             }
77             current = current.neste;
78         }
79         return ret.toArray();
80     }
81
82     @Override
83     public int antall(Sjanger sjanger) {

```

```

82     int telling = 0;
83     LinearNode<Film> current = head;
84     while (current != null) {
85         if (current.data.getSjanger() == sjanger){
86             telling++;
87         }
88         current = current.neste;
89     }
90     return telling;
91 }
92
93 @Override
94 public int antall() {
95     return antall;
96 }
97
98 public Film[] toArray(){
99     Film[] ret = new Film[antall];
100     LinearNode<Film> current = head;
101     int i = 0;
102     while (current != null) {
103         ret[i] = current.data;
104         current = current.neste;
105     }
106     return ret;
107 }
108
109 }

```

ein forskjell som må til når ein bytter implementasjon er å fjerne inparameteret når ein opretter nytt filmarkiv, sidan den nodebaserte implementasjonen ikkje har behov for bestemt lengde

3 Oppgåve 3

3.1 Deloppgåve 3 a)

i.

$$4n^2 + 50n - 10 = O(n^2)$$

ii.

$$10n + 4 \log_2 n + 30 = O(n)$$

iii.

$$13n^3 + 22n^2 + 50n + 20 = O(n^3)$$

iv.

$$35 + 13 \log_2 n = O(\log n)$$

3.2 Deloppgåve 3 b)

mengden tilordningar koden

```
1  sum = 0;
2  for (int i = n; i > 1; i = i/2) {
3      sum = sum + i;
4  }
```

gjør. Vil gå opp jo større n er, men siden i hopper med halvparten av i per loop, så vil det forholdsvis gå ned i kor mange tilordninger per størrelse på n . Så tidskompleksiteten blir $O(\log n)$

3.3 Deloppgåve 3 c)

```
1  sum = 0;
2  for (int i = 1; i <= n; i++) {
3      for (int j = 1; j <= n; j = j * 2) {
4          }
5      }
6  }
```

ytterste løkka er ei klassisk for løkke med $O(n)$ kompleksitet. Den innerste aukar med gange 2, så den $O(\log n)$ til sammen får vi då $O(n \log n)$ kompleksitet

3.4 Deloppgåve 3 d)

Areal er andregradsfunksjon så den blir $O(n^2)$ omkrets er førstegradsfunksjon, så den blir $O(n)$

3.5 Deloppgåve 3 f)

i.

$$8n + 4n^3 = O(n^3)$$

ii.

$$10 \log_2 n + 20 = O(\log n)$$

iii.

$$20n + 2n \log_2 n + 11 = O(n \log n)$$

iv.

$$4 \log_2 n + 2n = O(n)$$

rangert best til verst: ii \rightarrow iv \rightarrow iii \rightarrow i

3.6 Deloppgåve 3 g)

her er koden for målinga

```
1
2
3 public class oppg3g {
4     public static void main(String[] args) {
5
6         long n = (long) Math.pow(10,7);
7         long tidf r = System.currentTimeMillis();
8         tid(n);
9         long tidetter = System.currentTimeMillis();
10        System.out.println("10^7 bruker " + (long)(tidetter -
        tidf r) + " millisekund");
11
12        n*= 10;
13
14        tidf r = System.currentTimeMillis();
15        tid(n);
16        tidetter = System.currentTimeMillis();
17        System.out.println("10^8 bruker " + (long)(tidetter -
        tidf r) + " millisekund");
18
19        n*= 10;
20
21        tidf r = System.currentTimeMillis();
22        tid(n);
23        tidetter = System.currentTimeMillis();
24        System.out.println("10^9 bruker " + (long)(tidetter -
        tidf r) + " millisekund");
25
26        /*n*= 10;
27
28        tidf r = System.currentTimeMillis();
29        tid(n);
30        tidetter = System.currentTimeMillis();
31        System.out.println("10^10 bruker " + (long)(tidetter -
        tidf r) + " millisekund");
32        */
33    }
34
35    public static void tid(long n) {
36        long k = 0;
37        for (long i = 1; i <= n; i++) {
38            k = k + 5;
39        }
40    }
41 }
```

her er resultatet eg fikk etter å køyre den nokre gongar

| 10^7 | 10^8 | 10^9 |
|--------|--------|--------|
| 3ms | 4ms | 17ms |
| 0ms | 0ms | 17ms |
| 8ms | 9ms | 16ms |
| 4ms | 5ms | 18ms |
| 4ms | 4ms | 16ms |

vanskeleg å sjå hastigheita $T(n) = cn$ når 10^7 og 10^8 brukar ca. samme tid, så eg slengte på ein 10^{10} og fikk desse resultata

| 10^7 | 10^8 | 10^9 | 10^{10} |
|--------|--------|--------|-----------|
| 0ms | 8ms | 16ms | 164ms |
| 3ms | 0ms | 16ms | 172ms |
| 4ms | 4ms | 16ms | 163ms |
| 4ms | 3ms | 16ms | 173ms |
| 0ms | 0ms | 22ms | 165ms |

No ser det meir riktig ut sida den legger på ca. eit siffer per 10^x