DM502

Forelæsning 10

Indhold

- Rekursion
  - Fakultet $n!$
  - Fibonaccitallene
- 2. projektopgave
  - Opgaven
  - Formalia

Rekursion

Rekursion

- Prøv at definere "en liste af tal" uden at bruge ordet "liste"

Rekursion

- Prøv at definere "en liste af tal" uden at bruge ordet "liste"
- Svært, ik?

Rekursion

- Prøv at definere "en liste af tal" uden at bruge ordet "liste"
- Svært, ik?
- "En liste af tal er en lis... øhh... sekvens af tal?"

Rekursion

- Prøv at definere "en liste af tal" uden at bruge ordet "liste"
- Svært, ik?
- "En liste af tal er en lis... øhh... sekvens af tal?"
  - "Sekvens" er bare et andet ord for "liste"

Rekursion

- Prøv at definere "en liste af tal" uden at bruge ordet "liste"
- Svært, ik?
- "En liste af tal er en lis... øhh... sekvens af tal?"
  - "Sekvens" er bare et andet ord for "liste"
- Man behøver måske ikke undgå at bruge ordet "liste"

Rekursion

- Hvad med?
  - "En liste af tal er enten
    - et tal, eller
    - et tal efterfulgt af en liste af tal"
- Lister af tal
  - 42
    - Et tal
  - 7, 9, 13
    - Et tal efterfulgt af (et tal efterfulgt af (et tal))
    - 7 9 13
  - Osv...

Rekursion

- En liste af tal kan defineres i termer af sig selv
  - "Definitionen kaldes rekursiv"
  - "En liste af tal er defineret rekursivt"
- Rekursion
  - Når noget er defineret ved brug af sig selv

Rekursion

- Vi har set funktioner der kalder andre funktioner
  - Fx `main`-metoden der kalder `power(x, y)`
  - Osv...

Power eksempel

```
public class PowerExample {
    public static void main( String[] args ) {
        int result;
        int a, b;

        a = 2;
        b = 4;
        result = power( a, b );
        System.out.println( result );

        result = power( b, 0 );
        System.out.println( result );

        System.out.println( power( 1, 2 ) );
    }

    public static int power( int x, int y ) {
        int result = 1;
        int i;

        for( i = 1; i <= y; ++i ) {
            result = result * x;
        }

        return result;
    }
}
```

Rekursion

- Rekursion i programmering
  - Når en funktion kalder sig selv
- Fakultet
  - $5! = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$
  - $n! = n \cdot (n-1)!$

Rekursion

- Fakultet
  - Et første forsøg

```
public class Factorial {
    public static void main( String[] args ) {
        System.out.println( "5! = " + fac(5) );
    }

    public static int fac( int n ) {
        int n1 = fac( n-1 );
        return n * n1;
    }
}
```

Rekursion

- Fakultet
  - Et første forsøg

```
public class Factorial {
    public static void main( String[] args ) {
        System.out.println( "5! = " + fac(5) );
    }

    public static int fac( int n ) {
        int n1 = fac( n-1 );
        return n * n1;
    }
}
```

```
Terminal — bash — 66×8
mac-sdu-00001:10 petersk$ javac Factorial.java
mac-sdu-00001:10 petersk$ java Factorial
Exception in thread "main" java.lang.StackOverflowError
        at Factorial.fac(Factorial.java:12)
        at Factorial.fac(Factorial.java:12)
        at Factorial.fac(Factorial.java:12)
        at Factorial.fac(Factorial.java:12)
        at Factorial.fac(Factorial.java:12)
```

Rekursion

Rekursion

- Fakultet
  - Hvad er der galt!?!

Rekursion

- Fakultet
  - Hvad er der galt!?!
  - $fac(5) = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 \cdot 0 \cdot -1 \cdot -2 \cdot -3 \cdot \ldots$

Rekursion

- Fakultet
  - Hvad er der galt!?!
  - $fac(5) = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 \cdot 0 \cdot -1 \cdot -2 \cdot -3 \cdot \ldots$
  - Rekursionen stopper aldrig

Rekursion

- Fakultet
  - Hvad er der galt!?!
  - $fac(5) = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 \cdot 0 \cdot -1 \cdot -2 \cdot -3 \cdot \ldots$
  - Rekursionen stopper aldrig
  - Basistilfælde
    - $1! = 1$

Rekursion

- Fakultet
  - Et nyt forsøg

```
public class Factorial {
    public static void main( String[] args ) {
        System.out.println( "5! = " + fac(5) );
    }

    public static int fac( int n ) {
        int n1;
        if( n == 1 ) {
            return 1;
        } else {
            n1 = fac( n-1 );
            return n * n1;
        }
    }
}
```

Rekursion

- Fakultet
  - Et nyt forsøg

```
public class Factorial {
    public static void main( String[] args ) {
        System.out.println( "5! = " + fac(5) );
    }

    public static int fac( int n ) {
        int n1;
        if( n == 1 ) {
            return 1;
        } else {
            n1 = fac( n-1 );
            return n * n1;
        }
```

```
Terminal — bash — 66×5
mac-sdu-00001:10 petersk$ javac Factorial.java
mac-sdu-00001:10 petersk$ java Factorial
5! = 120
mac-sdu-00001:10 petersk$
```

Rekursion

- Fakultet
  - Er det så et godt program?

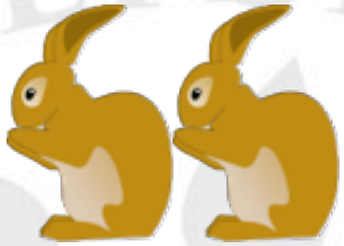
Fibonacci-tallene

- Introduceret i vesten af Leonardo af Pisa (Fibonacci) i en bog fra 1202
  - Kendt tidligere og kommer oprindeligt fra Indien
- Betragt en (urealistisk) kanin-population
- Der starter med at være et kanin-par
- Et nyt kanin-par kan ikke få unger, før de er 1 måned gammel
- Et kanin-par kan få to unger på 1 måned

Fibonacci-tallene

Fibonacci-tallene

Fibonacci-tallene

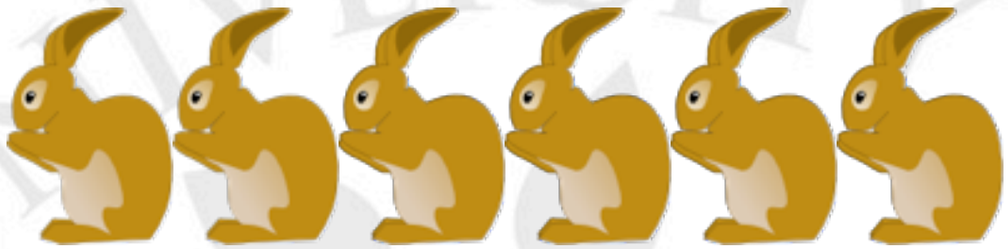


Fibonacci-tallene

Fibonacci-tallene

Fibonacci-tallene

Fibonacci-tallene



Fibonacci-tallene

Fibonacci-tallene

Fibonacci-tallene

Fibonacci-tallene

- 1. måned - 1 kaninpar
- 2. måned - 1 kaninpar
- 3. måned - 2 kaninpar
- 4. måned - 3 kaninpar
- 5. måned - 5 kaninpar
- 6. måned - 8 kaninpar
- 7. måned - ?

Fibonacci-tallene

Fibonacci-tallene

- Måned n
  - (#kaniner i måned n-1) + (#kaniner i måned n-2)
  - Alle kaniner der var der i måned n-1 er der også i måned n
  - Alle kaniner der var der i måned n-2 får unger i måned n

Fibonacci-tallene

- Måned n
  - (#kaniner i måned n-1) + (#kaniner i måned n-2)
  - Alle kaniner der var der i måned n-1 er der også i måned n
  - Alle kaniner der var der i måned n-2 får unger i måned n
- Lad f(n) være antallet af kaniner i måned n
  - f(n) = f(n-1) + f(n-2)
  - f(n) - det n'te fibonacci-tal
  - f(1) = 1, f(2) = 1, f(3) = 2, f(4) = 3, f(5) = 5, f(6) = 8, osv...
  - Bemærk definitionen er rekursiv f(n) afhænger af f(n-1) og f(n-2)

1. forsøg

```
public class Fibonacci {
    public static void main( String[] args ) {
        System.out.println( f(6) );
    }

    public static int f( int n ) {
        return f(n-1) + f(n-2);
    }
}
```

1. forsøg

```
public class Fibonacci {
    public static void main( String[] args ) {
        System.out.println( f(6) );
    }

    public static int f( int n ) {
        return f(n-1) + f(n-2);
    }
}
```

```
Terminal — bash — 66×8
mac-sdu-00001:10 petersk$ javac Fibonacci.java
mac-sdu-00001:10 petersk$ java Fibonacci
Exception in thread "main" java.lang.StackOverflowError
        at Fibonacci.f(Fibonacci.java:7)
        at Fibonacci.f(Fibonacci.java:7)
        at Fibonacci.f(Fibonacci.java:7)
        at Fibonacci.f(Fibonacci.java:7)
        at Fibonacci.f(Fibonacci.java:7)
        at Fibonacci.f(Fibonacci.java:7)
```

Fibonacci-tallene

- Samme fejl som sidst
  - Mangler basis-tilfælde
  - Hmm....
  - Vi ved at f(1) = 1

2. forsøg

```
public static int f( int n ) {
     if( n == 1 )
          return 1;
     else
          return f(n-1) + f(n-2);
}
```

2. forsøg

```
public static int f( int n ) {
     if( n == 1 )
          return 1;
     else
          return f(n-1) + f(n-2);
}
```

```
Terminal — bash — 66×8
mac-sdu-00001:10 petersk$ javac Fibonacci.java
mac-sdu-00001:10 petersk$ java Fibonacci
Exception in thread "main" java.lang.StackOverflowError
        at Fibonacci.f(Fibonacci.java:7)
        at Fibonacci.f(Fibonacci.java:10)
        at Fibonacci.f(Fibonacci.java:10)
        at Fibonacci.f(Fibonacci.java:10)
        at Fibonacci.f(Fibonacci.java:10)
```

Fibonacci-tallene

- Stadig fejl...?!?
  - Lad os prøve at debugge det i JSwat

Fibonacci-tallene

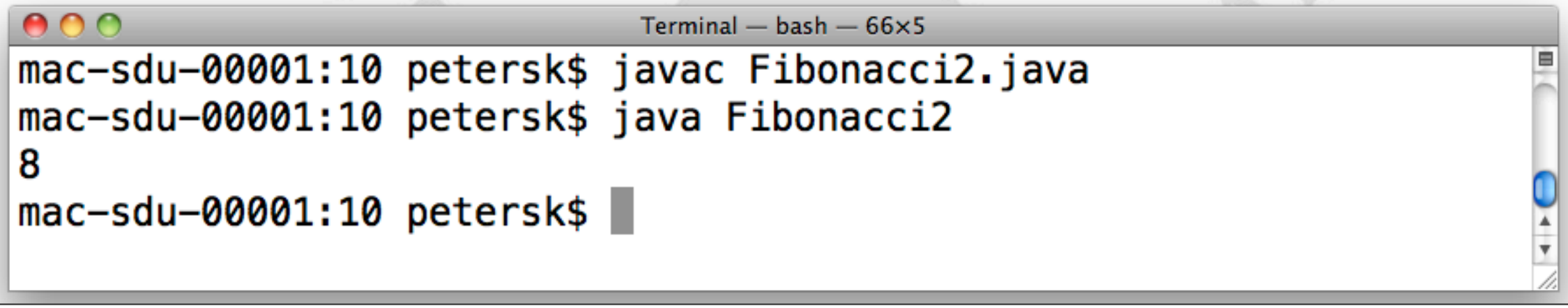
- Stadig fejl...?!?
  - Lad os prøve at debugge det med System.out.println
  - Hmmm...
  - f(2) burde give 1
  - Men programmet giver
    - f(2) = f(1) + f(0)
      = 1 + f(0)
  - Ups...

3. forsøg

```
public static int f( int n ) {
     if( n == 1 )
          return 1;
     else if( n == 2 )
          return 1;
     else
          return f(n-1) + f(n-2);
}
```

3. forsøg

```
public static int f( int n ) {
     if( n == 1 )
          return 1;
     else if( n == 2 )
          return 1;
     else
          return f(n-1) + f(n-2);
}
```



Fibonacci-tallene

- Fibonacci-tallene kan ses mange steder i naturen
  - Solsikker

  - Grankogler

  - osv...
- Tæt forbundet med det gyldne snit
  - f(n+1) / f(n) -> det gyldne snit for n -> uendelig

Rekursion

- Recursion: If you still don't get it, See: "Recursion".

2. delprojektopgave

Overblik

Overblik

- Finde nøglen til en tekst, der er krypteret med Caesar
  - Indlæs en krypteret tekst fra en fil
  - Indlæs hyppighedsfordelingen fra en fil
  - Brug en HashMap til at repræsentere hyppighedsfordelinger
  - Find hyppighedsfordelingen for den inputteksten
  - Sammenlign de to fordelinger
  - Bestem nøglen

Overblik

- Finde nøglen til en tekst, der er krypteret med Caesar
  - Indlæs en krypteret tekst fra en fil
  - Indlæs hyppighedsfordelingen fra en fil
  - Brug en HashMap til at repræsentere hyppighedsfordelinger
  - Find hyppighedsfordelingen for den inputteksten
  - Sammenlign de to fordelinger
  - Bestem nøglen
  - Udskriv den dekrypterede tekst til skærmen (og fil)

Overblik

- Finde nøglen til en tekst, der er krypteret med Caesar
  - Indlæs en krypteret tekst fra en fil
  - Indlæs hyppighedsfordelingen fra en fil
  - Brug en HashMap til at repræsentere hyppighedsfordelinger
  - Find hyppighedsfordelingen for den inputteksten
  - Sammenlign de to fordelinger
  - Bestem nøglen
  - Udskriv den dekrypterede tekst til skærmen (og fil)

Overblik

- Finde nøglen til en tekst, der er krypteret med Caesar
  - Indlæs en krypteret tekst fra en fil
  - Indlæs hyppighedsfordelingen fra en fil
  - Brug en HashMap til at repræsentere hyppighedsfordelinger
  - Find hyppighedsfordelingen for den inputteksten
  - Sammenlign de to fordelinger
  - Bestem nøglen
  - Udskriv den dekrypterede tekst til skærmen (og fil)

- Skal arbejde korrekt med inputteksterne, der er tilgængelig fra hjemmesiden

engelsk2.crypt

Puzaybjavyz jhu zluk lthps av hss vy zlsljalk pukpcpkbhs Bzlyz, Zabkluaz, Nyvbwz, Alhjopun Hzzpzahuaz, Puzaybjavyz vy Vizlyclyz pu h Jvbyzl. Myvt h Ishjrivhyk Slhyu jvbyzl, lthpsz jhuuva il zlua av hufvul dov pz uva h tltily vm aol jvbyzl.

engelsk.dat

```
a  8.167
b  1.492
c  2.782
d  4.253
e 12.702
f  2.228
g  2.015
h  6.094
i  6.966
j  0.153
k  0.772
l  4.025
m  2.406
n  6.749
o  7.507
p  1.929
...
```

engelsk2.crypt

Puzaybjavyz jhu zluk lthps av hss vy zlsljalk pukpcpkbhs Bzlyz, Zabkluaz, Nyvbwz, Alhjopun Hzzpzahuaz, Puzaybjavyz vy Vizlyclyz pu h Jvbyzl. Myvt h Ishjrivhyk Slhyu jvbyzl, lthpsz jhuuva il zlua av hufvul dov pz uva h tltily vm aol jvbyzl.

Instructors can send email to all or selected individual Users, Students, Groups, Teaching Assistants, Instructors or Observers in a Course. From a Blackboard Learn course, emails cannot be sent to anyone who is not a member of the course.

Formalia

- Regler - de samme som sidst
- Afleveringsfrist
  - Mandag d. 1. november kl. 12:00
  - Dvs. ca. 4 uger
  - Dvs. meget god tid

- Filerne engelsk.dat, engelsk1.crypt, engelsk2.crypt og engelsk3.crypt er tilgængelig på hjemmesiden

- Følg med på hjemmesiden!

God fornøjelse!