Tölvunarfræði 1
Fyrirlestur 17: Endurkvæmni I

Hjálmtýr Hafsteinsson

Haust 2015

HÁSKÓLI ÍSLANDS
IÐNAÐARVERKFRÆÐI-, VÉLAVERKFRÆÐI-
OG TÖLVUNARFRÆÐIDEILD

Í síðasta fyrirlestri

- Einingaprófun (*unit testing*)
- Stærri dæmi um forritasöfn
  - Gagnagreining (**`StdStats`**)
  - Inntak/úttak fyrir fylki (**`StdArrayIO`**)
  - Ítruð föll (**`IFS`**)
- Einingaforritun (*modular programming*)

Kafli 2.2

Í þessum fyrirlestri

- Endurkvæmni (*recursion*)
- Dæmi um endurkvæmni
  - **`Factorial`**
  - Reiknirit Evklíðs
  - Fibonacci tölur
- Eiginleikar endurkvæmni

Kafli 2.3

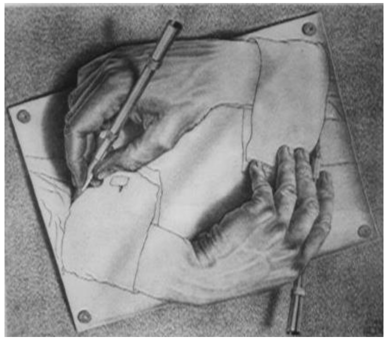
Endurkvæmni (*recursion*)

- Endurkvæm skilgreining þýðir að hugtak er skilgreint með því að nota hugtakið sjálft
- Endurkvæm forritun felst í að fall kalli á sjálft sig til að leysa verkefni

- Ekki nauðsynlegt að nota endurkvæmni
  - Endurkvæmar lausnir eru oft styttri og augljósari en aðrar

Hluti af verkfærakistu nútíma forritara

Fáum dýpri skilning á eðli forritunar



Einfalt dæmi: `Factorial`

- Fyrsta endurkvæma forritið er oftast að reikna n! (n hrópmerkt, *n factorial*):

```
public static int factorial(int n) {
   if (n == 0)  return 1;
   else         return n * factorial(n-1);
}
```

Forrit fyrir n!

$1! = 1$

$n! = n*(n-1)!$

Skilgreining á n!

- Endurkvæmni hefur tvo hluta:
  - Grunntilfelli (*base case*): $1! = 1$
  - Almenna tilfellið (*general case*): $n! = n*(n-1)!$

Factorial.java

(Java Visualizer)

```
public class Factorial {

   public static int factorial(int n) {
      if (n == 0) return 1;
      else        return n * factorial(n-1);
   }

   public static void main(String[] args) {
      int N = Integer.parseInt(args[0]);
      StdOut.println(factorial(N));
   }
}
```

Endurkvæma fallið

`main`-fall sem kallar á `factorial`

```
% java Factorial 5
120

% java Factorial 10
3628800

% java Factorial 13
1932053504
```

n! stækkar mjög hratt

Hér er orðið yfirflæði í `int` breytunni

Skoða keyrslu á `Factorial`

```
public static int factorial(int n, int t) {
   String indent = spaces(3*t);
   StdOut.println(indent + "factorial(" + n + ")");
   int fact;

   if (n == 0) fact = 1;
   else        fact = n * factorial(n-1, t+1);

   StdOut.println(indent + "return " + fact);
   return fact;
}
```

Nú eru tvö viðföng

Viðfangið `t` segir hversu djúpt endurkvæmnin er komin

```
% java FactorialTrace 4
factorial(4)
   factorial(3)
      factorial(2)
         factorial(1)
            factorial(0)
            return 1
         return 1
      return 2
   return 6
return 24
24
```

Prentum "`factorial(n)`" þegar við förum inn í fallið

Prentum "`return x`" þegar við förum út úr fallinu

Nokkrir punktar um `Factorial`

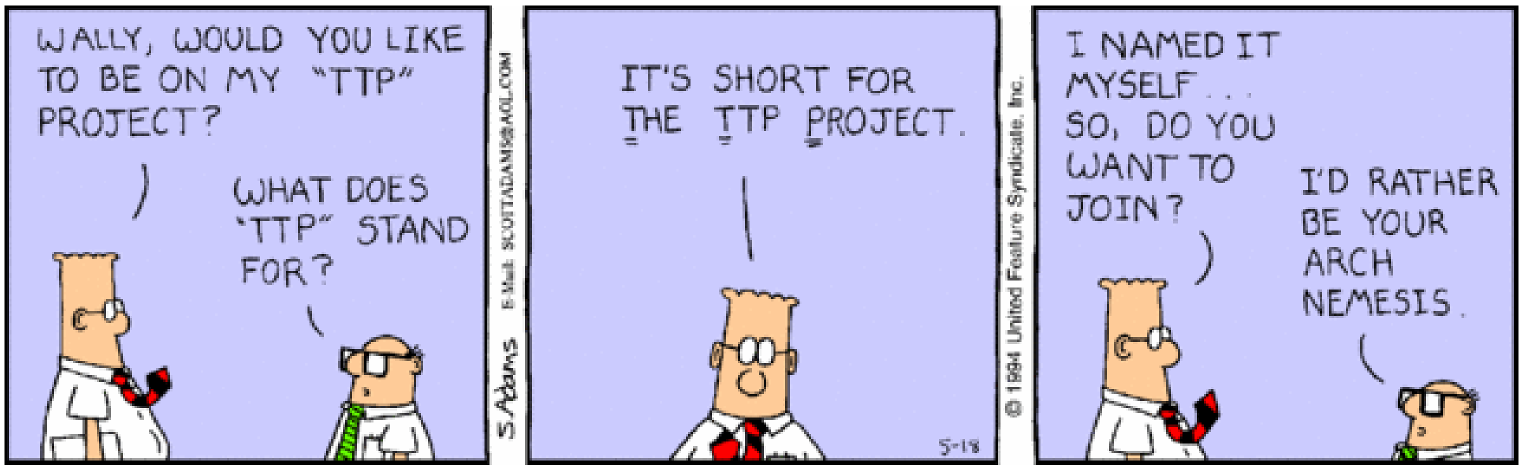
- Mikilvægt að hafa grunntilvik
  - Lendum annars í endalausri endurkvæmni
  - Fáum Java villuna `java.lang.StackOverflowError`
    - Það er tekið frá minni fyrir hvert endurkvæmt kall og það klárast

- Getum skrifað forritið án endurkvæmni:

```
public static int fact(int n) {
  int f = 1;
  for (int i=1; i<=n; i++)
     f *= i;
  return f;
}
```

Ekki eins líkt skilgreiningunni á *n*!

Endurkvæmni í Dilbert



– TTP stendur fyrir
  - "The TTP Project"
  - eða "The (The TTP Project) Project"
  - eða "The (The (The TTP Project) Project) Project" ...

Vantar grunntilfelli, svo þetta er óendanleg endurkvæmni

Endurkvæmni á ýmsum stöðum

- Nafnið VISA stendur fyrir "VISA International Service Association"
  - Þetta er óendanleg endurkvæm skilgreining
- Ríkisborgarararéttur:
  - Einstaklingur er íslenskur ríkisborgari ef
    - hann hefur sótt um og fengið ríkisborgararétt
    - foreldrar hans eru íslenskir ríkisborgarar
- Brandari:
  - Til að skilja endurkvæmni þarftu að skilja endurkvæmni

Reyndar er þetta gervi-skammstöfun! VISA þýðir ekki neitt

Endurkvæmni og þrepun

- Stærðfræðileg þrepun (*mathematical induction*) er aðferð til að sanna setningar
- Sanna að setning gildi um jákvæðu heiltöluna *N*:
  - Grunntilfelli: Sýna að setningin sé sönn fyrir ákveðið gildi, oftast $N = 0$ eða 1
  - Almennt tilfelli: Ef satt fyrir allar jákvæðar heiltölur $< N$, þá er setningin sönn fyrir *N*
- Mjög svipuð uppsetning og í endurkvæmni
  - Setningar sem hafa þrepunarsönnun er oftast auðvelt að reikna út með endurkvæmu forriti

Annað dæmi

- Getum notað endurkvæmni til að reikna *N*-tu þýðtöluna (*harmonic number*)

```
public static double H(int n) {

   if (n == 1) return 1.0;

   return H(n-1) + 1.0/n;
}
```

Ath.: Hér er almenna tilfellið ekki í `else`-hluta

Þegar endurkvæma kallið er í síðustu skipun fallsins þá kallast það <u>halaendurkvæmni</u> (*tail recursion*)

Auðvelt að breyta halaendurkvæmnum föllum yfir í ítrun

Fyrirlestraræfing

1. Hvað gerist ef kallað er á `factorial`-fallið með neikvæðri tölu?

2. Búið til óendanlega endurkvæma skammstöfun tengda þessu námskeiði (svipað og TTP)

3. Skrifið endurkvæmt fall sem reiknar 1+2+ ... +N
   (Þetta skilar reyndar tölunni $N(N+1)/2$)

Breyta ítrun yfir í halaendurkvæmni

- Fallið `finna(int[] a, int x)` leitar að **x** í fylkinu **a**:

```
public static int finna(int[] a, int x) {
   for (int i=0; i<a.length; i++)
      if (a[i] == x) return i;
   return -1;
}
```

(Java Visualizer)

- Breytum því í halaendurkvæmt fall:

Þurfum þá að senda núv. staðsetningu sem viðfang

```
public static int finnaend(int[] a, int i, int x) {
   if (i >= a.length) return -1;
   if (a[i] == x) return i;
   return finnaend(a, i+1, x);
}
```

```
...
StdOut.println(finnaend(a, 0, 5));
...
```

HÁSKÓLI ÍSLANDS
IÐNAÐARVERKFRÆÐI-, VÉLAVERKFRÆÐI- OG TÖLVUNARFRÆÐIDEILD

Stærsti samdeilir

- Stærsti samdeilir (*greatest common divisor*, *gcd*) tveggja heiltalna er stærsta heiltala sem gengur uppí þær báðar
- Dæmi:

gcd(4032, 1272) = 24

$$4032 = 2^6 \times 3^2 \times 7^1$$
$$1272 = 2^3 \times 3^1 \times 53^1$$
$$\text{gcd} = 2^3 \times 3^1 = \mathbf{24}$$

Frumþáttun talnanna

Stærsti samdeilir er margfeldi sameiginlegra frumþátta

Notkun:
Til að stytta almenn brot
Hluti af RSA dulkóðunaraðferðinni

Reiknirit Evklíðs

- Gríski stærðfræðingurinn Evklíð (~300 f.K.) setti fram aðferð til að finna stærsta samdeili

$$\gcd(p,q) = \begin{cases} p & \text{ef } \; q = 0 \\ \gcd(q, p\%q) & \text{annars} \end{cases}$$

Grunntilfellið

Almenna tilfellið

```
gcd(4032, 1272)  = gcd(1272, 216)
                 = gcd(216, 192)
                 = gcd(192, 24)
                 = gcd(24, 0)
                 = 24.
```

4032 = 3 x **1272** + **216**

1272 = 5 x **216** + **192**

216 = 1 x **192** + **24**

192 = 8 x **24** + **0**

24 = 0 x **0** + **0**

Reikninrit Evklíðs í Java

- Endurkvæm útgáfa:

([Java Visualizer])

```
public static int gcd(int p, int q)
{
   if (q == 0) return p;
   return gcd(q, p % q);
}
```

- Óendurkvæm útgáfa:

```
public static int gcd2(int p, int q) {
   while (q != 0) {
      int temp = q;
      q = p % q;
      p = temp;
   }
   return p;
}
```

Víxlum sjálf á `p` og `q`

Ekki eins skýr útgáfa

Fibonacci tölur

- Þraut:
  - Eitt kanínupar er sett í girðingu. Hvert par eignast eitt par af kanínum þegar þær eru kynþroska, sem er eftir 1 mánuð. Hvað verða margar kanínur í girðingunni eftir 1 ár?
    - Í lok fyrsta mánaðar er aðeins 1 par
    - Í lok annars mánaðar eignast þær 1 par, svo þá eru 2 pör
    - Í lok þriðja mánaðar eignast upphaflega parið 1 par, svo þá eru 3 pör
    - Í lok fjórða mánaðar eignast tvö pörin ný pör, svo þá eru 5 pör
    - ...
    - Almennt: Ef x_n pör á tíma n þá á tíma n+1 eru áfram x_n pör + ungar þeirra para sem voru til á tíma n-1, sem eru x_{n-1}

Formúla: $x_{n+1} = x_n + x_{n-1}$

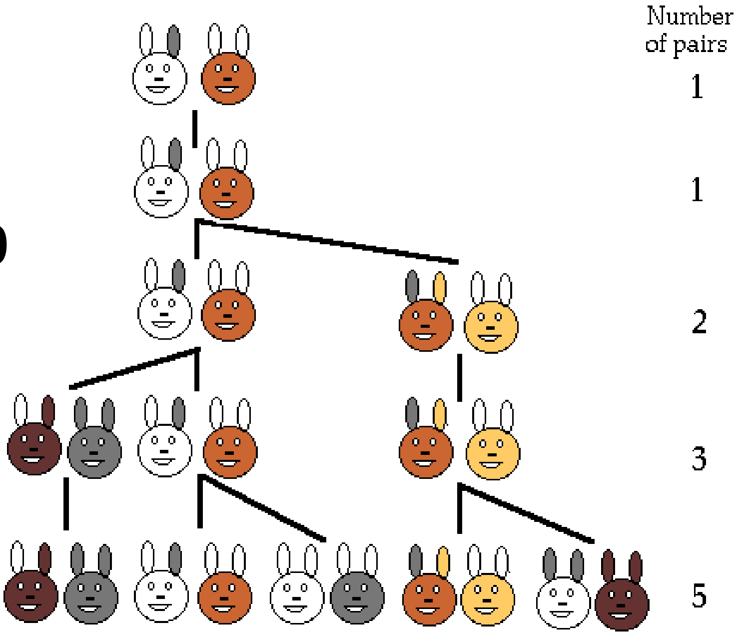
Fibonacci tölur

L. P. Fibonacci
(1170 - 1250)

- Fjöldi kanínupara verður 1, 1, 2, 3, 5, 8, 13, ...
- Fáum endurkvæmu skilgreininguna:

$$F(n) = \begin{cases} 0 & \text{ef } n = 0 \\ 1 & \text{ef } n = 1 \\ F(n-1) + F(n-2) & \text{annars} \end{cases}$$

- Fibonacci tölur koma mjög víða upp
- Hafa marga [áhugaverða eiginleika]



HÁSKÓLI ÍSLANDS
IÐNAÐARVERKFRÆÐI-, VÉLAVERKFRÆÐI- OG TÖLVUNARFRÆÐIDEILD

Forrit fyrir Fibonacci tölur

- Virðist augljóst að nota endurkvæmni til að finna *N*-tu Fibonacci töluna:

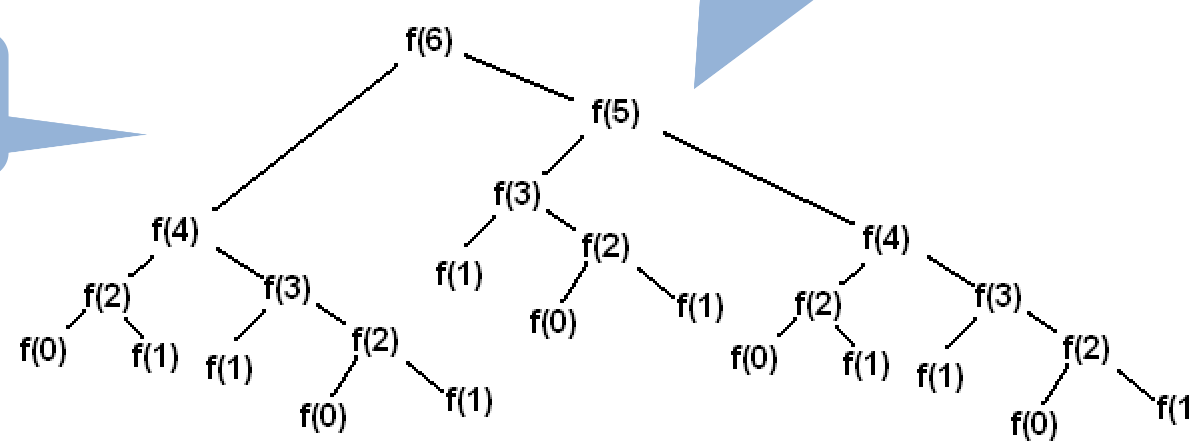
(Java Visualizer)

```
public static long fib(int n) {
    if (n == 0) return 0;
    if (n == 1) return 1;
    return fib(n-1) + fib(n-2);
}
```

En þetta er mjög óhagkvæm aðferð til að finna Fibonacci tölur

f(4) er reiknað tvisvar, f(3) þrisvar, f(2) reiknað fimm sinnum, o.s.frv.

Erum sífellt að reikna sömu tölurnar aftur og aftur



Betra forrit fyrir Fibonacci

- Reikna Fibonacci tölurnar inn í fylki:

(Java Visualizer)

```
public static long fibo(int n) {
    long[] f = new long[n+1];
    f[1] = 1;
    for (int i=2; i<=n; i++) {
        f[i] = f[i-1] + f[i-2];
    }
    return f[n];
}
```

Búum til fylki til að setja Fibonacci tölurnar í

`f[0] = 0` og `f[1] = 1`

Notum Fibonacci formúluna, nema hér er ekki endurkvæmni

```
public static long fibon(int n) {
    long f1 = 0, f2 = 1;
    for (int i=2; i<=n; i++) {
        long f3 = f1 + f2;
        f1 = f2;
        f2 = f3;
    }
    return f2;
}
```

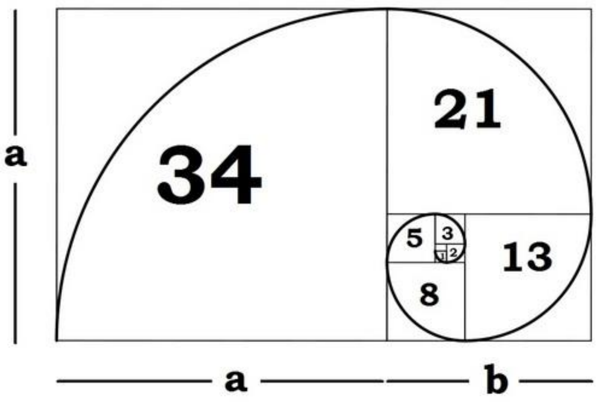
Sleppum fylkinu, geymum tvö síðustu gildin í breytunum `f1` og `f2`

Besta forritið fyrir Fibonacci tölur

- Reyndar til lokuð formúla fyrir Fibonacci tölur!
  - F_n er næsta heiltala við $\dfrac{\varphi^n}{\sqrt{5}}$

    þar sem $\varphi = \dfrac{1+\sqrt{5}}{2} \approx 1.6180339887\cdots$

    er gullna hlutfallið (*golden ratio*)

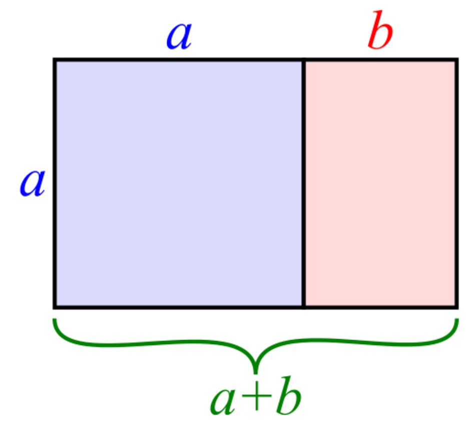



```
public static long fibona(int n) {
   double phi = (1 + Math.sqrt(5)) / 2.0;
   return Math.round(Math.pow(phi, n) / Math.sqrt(5));
}
```

Engin lykkja, bara hrein útreikningur á formúlu

Gullna hlutfallið φ

Hlutfall er gullið ef $\frac{a+b}{a} = \frac{a}{b}$



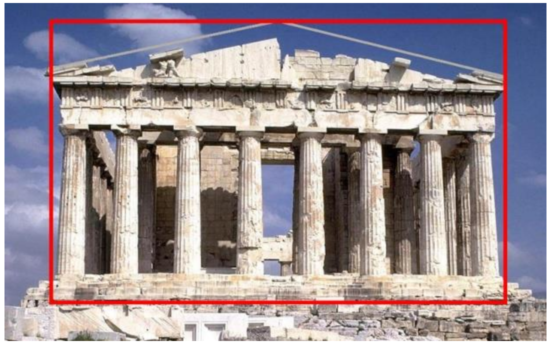
Hlutfallið í A4 pappír er √2 ≈ 1.41421...

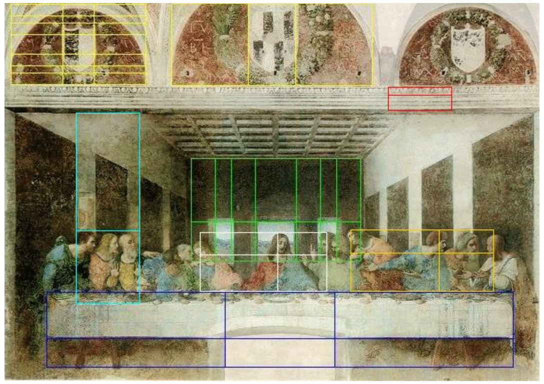
Hlutfall hliðstæðra Fibonacci talna nálgast *φ:*

3/2 = 1.5
5/3 = 1.6666...
8/5 = 1.6
13/8 = 1.625
21/13 = 1.61538...

$$\boxed{\lim_{n\to\infty} \frac{F_{n+1}}{F_n} = \varphi}$$

Kemur víða upp í arkitektúr og listum:





Fyrirlestraræfing

4. Reiknið gcd(15, 6) með aðferð Evklíðs

5. Ef kanínurnar væru 2 mánuði að verða kynþroska þá væri $g_0 = 0$, $g_1 = 1$, $g_2 = 1$ og síðan er $g_n = g_{n\text{-}1} + g_{n\text{-}3}$. Reiknið fyrstu 10 tölurnar í þessari runu

6. Skrifið endurkvæmt fall **`stjornur(int n)`**, sem skilar streng sem samanstendur af **`n`** stjörnum (**`*`**)
   Vísbending: Skeytið einni stjörnu framan við **`stjornur(n-1)`**

Samantekt

- Í þessum tíma:
  - Endurkvæmni
  - Nokkur einföld dæmi

Kafli 2.3

- Í næsta tíma:
  - Endurkvæmni í myndum
  - Brotamyndir (*fractals*)

Kafli 2.3