TÖL101G - Tölvunarfræði 1

Vikublað 3

Lausn

Æfingar

1.3.9

```
public class RandomN {
    public static void main(String[] args) {
        int N = Integer.parseInt(args[0]);

        double sum = 0.0; // to keep track of average
        double x;
        for (int i = 0; i < N ; i++) {
            x = Math.random();
            System.out.println(x);
            sum += x;
        }
        // compute average as sum/N
        System.out.println("Average: " + sum/N);
    }
}
```

1.3.11

```
public class FunctionTable {
    public static void main(String[] args) {
        // tab characters are encoded as \t
        System.out.println("log(N)\tN\tN log(N)\tN^2\t\tN^3\t\t2^N");
        for (int N = 16; N <= 2048; N *= 2) {
            // This gives the base-2 logarithm of N
            int logN = (int) Math.round(Math.log(N)/Math.log(2.0));
            int twotoN = (int) Math.round(Math.pow(2.0,N));
            System.out.print(logN + "\t"+N + "\t" + N*logN + "\t\t");
            System.out.println(N*N + "\t\t" + N*N*N + "\t\t"+twotoN);
        }
    }
}
```

```
log(N) N N log(N) N^2 N^3 2^N
4 16 64 256 4096 65536
5 32 160 1024 32768 0
6 64 384 4096 262144 -1
7 128 896 16384 2097152 -1
8 256 2048 65536 16777216 -1
9 512 4608 262144 134217728 -1
```

```
10 1024 10240 1048576 1073741824 -1
11 2048 22528 4194304 0 -1
```

1.3.27

Hér er trixið að fá forritið til að skrifa `*` og stafabil í annan hvern staf og skipta um fyrsta staf í hverri línu. Einfaldast er að skoða summuna `i+j` og hvort hún er slétt eða oddatala með `i+j % == 0`.

```
public class CheckerBoard {
    public static void main(String[] args) {
        int N = Integer.parseInt(args[0]);

        for (int i = 1; i <= N; i++) {
            for (int j = 1; j <= N; j++) {
                if ((i+j) % 2 == 0) {
                    System.out.print(" ");
                } else {
                    System.out.print("*");
                }
            }
            System.out.println();
        }
    }
}
```

```
% java CheckerBoard 20
 * * * * *
* * * * *
 * * * * *
* * * * *
 * * * * *
* * * * *
 * * * * *
* * * * *
 * * * * *
* * * * *
```

Cloudcoder æfingar

1.3.14

```
public class PowersOfTwo {
    public static void main(String[] args) {
        int N = Integer.parseInt(args[0]);
        if (N>0) {
            int t = 1;
            while (t <= N/2) {
```

```
                t *= 2;
            }
            System.out.println(t);
        } else {
            System.out.println();
        }
    }
}
```

1.3.19

Hér fylgjum við lausninni fyrir Binary.java sem gefin var í bókinni og það sem komið var inn á cloudcoder. Eina trixið er að átta sig á hvaða staf á að prenta. Við vitum að þegar við eigum að prenta út l-ta stafinn er $v = k^l$ og $n < k^{l+1} = k \cdot v$ svo að l-ti stafurinn fæst með heiltöludeilingunni sem gefin er í lausn 1. Til að uppfæra n þarf að draga frá það gildi sem búið er að prenta út sem verður $c \cdot v$, þetta má líka gera með mod aðgerðinni. Í lausn 2 í kóðanum er í raun og veru verið að reikna heiltöludeilingu án þess að nota `/` og í stað `%` er notaður frádráttur. Að lokum þarf að prenta út stafinn, ef hann er ekki tala þá er hægt að nota if setningu til að finna gildið eða switch (bls. 70-71 í bók) sem er knappari.

```
public class Kary {
  public static void main(String[] args) {
    long i = Long.parseLong(args[0]);
    int k = Integer.parseInt(args[1]);

    long v = 1;

    while (v <= i/k) {
      v *= k;

    }

    long n = i;
    while (v > 0) {
      int c;
      // Lausn 1
      //c = (int) (n / v);
      //n = n % v;

      // Lausn 2
      for (int j = 0; j < k; j++) {
        if (n < (j+1)*v) {
          c = j;
          n -= j*v;
          break;
        }
      }
```

```
      v = v/k;

      if (c < 10) {
        System.out.print(c);
      } else {
        switch (c)
        {
          case 10: System.out.print('A'); break;
          case 11: System.out.print('B'); break;
          case 12: System.out.print('C'); break;
          case 13: System.out.print('D'); break;
          case 14: System.out.print('E'); break;
          case 15: System.out.print('F'); break;
        }
      }
    }
    System.out.println();
  }
}
```

Verkefni

1.3.34

Þetta er ekki hraðvirkasta leiðin til að telja prímtölurnar, en það má hraða forritinu með því að sleppa óþarfa aðgerðum. Sem dæmi þá er 2 prímtala og allar aðrar prímtölur eru oddatölur. Því teljum við 2 sér og skoðum síðan allar oddatölur. Til að sjá hvort oddatala i sé prímtala skoðum við hvort einhver önnur tala $1 < j < i$ gangi upp í i. Til þess þarf j að vera oddatala og ef j gengur upp í i þá gengur líka $\frac{i}{j}$ upp í i. Því er nóg að skoða minni töluna af j og $\frac{i}{j}$. Þar sem $j \cdot \frac{i}{j} = i$ þá verður minni talan

$$\min\{j, \frac{i}{j}\} \leq \sqrt{i}$$

```
public class PrimeCounter {
    public static void main(String[] args) {
        int N = Integer.parseInt(args[0]);

        int numPrimes = 0;

        // 2 is a prime
        numPrimes++;

        // only need to check odd numbers >= 3
        for (int i = 3; i <= N; i += 2) {
            boolean isPrime = true;
            int sqrt_i = (int) Math.sqrt(i);
```

```
            // check all odd numbers from 3 to square root of i if
            // they divide i
            for (int j = 3; j <= sqrt_i; j += 2) {
                if (i % j == 0) {
                    isPrime = false;
                    break;
                }
            }
            if (isPrime) {
                numPrimes++;
            }
        }

        System.out.print("There are " + numPrimes);
        System.out.println(" prime numbers between 1 and " + N);
    }
}
```

Með því að nota time skipunina (til á linux og mac) þá sjáum við hversu langan tíma það tekur að keyra forritið, 4.125 sekúndur á fartölvunni minni.

```
% time java PrimeCounter 10000000
There are 664579 prime numbers between 1 and 10000000

real 0m4.125s
user 0m4.107s
sys 0m0.062s
```