Dæmi 1:

```
public class fylki3
{
   private static String[] aux;
   public static void merge(String[] a, int lo, int mid, int hi)
   {
     int i = lo, j = mid+1;
     for (int k = lo; k <= hi; k++)
        aux[k] = a[k];
     for (int k = lo; k <= hi; k++)
     {
        if      (i > mid)               a[k] = aux[j++];
        else if (j > hi )               a[k] = aux[i++];
        else if (aux[j].compareTo(aux[i]) < 0) a[k] = aux[j++];
        else                            a[k] = aux[i++];
     }
   }
   public static void sort(String[] a)
   {
      aux = new String[a.length];
      sort(a, 0, a.length - 1);
   }
   private static void sort(String[] a, int lo, int hi)
   {
        if (hi <= lo) return;
        int mid = lo + (hi - lo)/2;
        sort(a, lo, mid);
        sort(a, mid+1, hi);
        merge(a, lo, mid, hi);
  }

  public static String finna(String [] a, String [] b, String [] c)
  {
    sort(a);
    sort(b);
    sort(c);
    int i = 0;
    int j = 0;
    int k = 0;
    while(i < a.length)
    {
      while(a[i].compareTo(b[j]) > 0 && j < b.length)
      {
        j++;
      }
      if(a[i].equals(b[j]))
      {
        while(a[i].compareTo(c[k]) > 0 && k < c.length)
        {
          k++;
        }
        if(a[i].equals(c[k]))
        {
          return a[i];
        }
      }
      i++;
    }
    return "-1";
  }
}
```

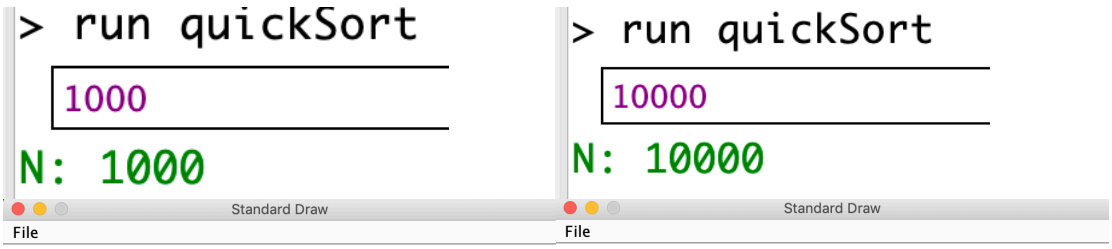
Dæmi 2:
Nlog_3(N) þar sem rétt eins og ef maður skiptir í tvennt er það
Nlog_2(N) þar sem ef það eru t.d. 8 stök skiptiru 3, log_2(8) = 3, og ef maður er með 9 stök og skiptir í þrennt skiptir maður aðeins 2, log_3(9) = 2.

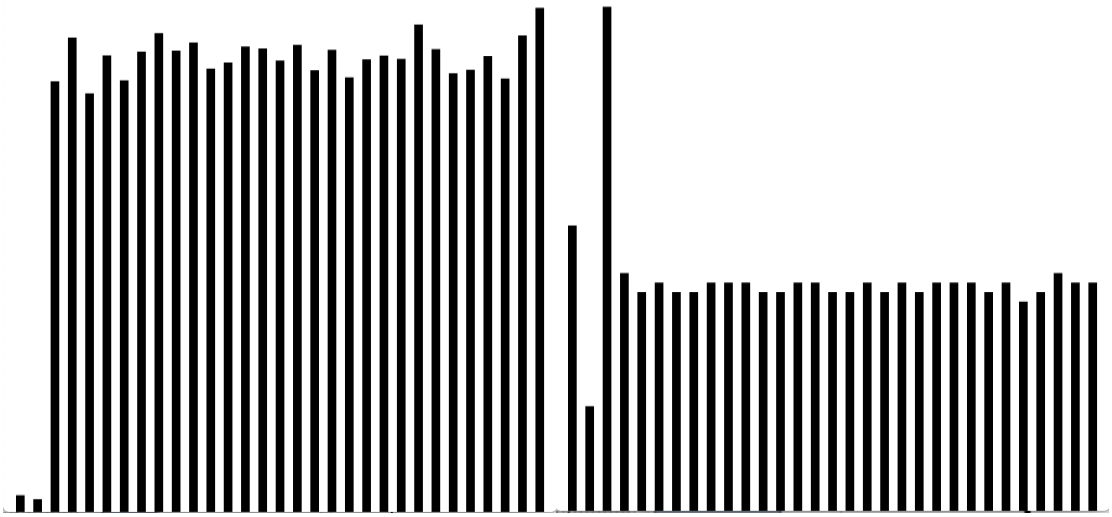
Dæmi 3:
Fyrsta tilvik:
[1,2,3,4,5,6,7,8,9,10]. Þá er fyrsta partition-ið á minnstu tölunni og þá gerist ekkert þar sem allar tölurnar eru stærri en 1 en það þurfti að bera saman n-2 sinnum, svo er partion-ið á næst minnstu og ekkert gerist heldur, þar sem allar tölurnar vinstra megin eru minni og þurfti að bera saman n-2 sinnum o.s.frv. Þannig það eru n-1 samanburðir n sinnum þ.e. n(n-2) sem er O(n^2).
Annað tilvik:
[10,9,8,7,6,5,4,3,2,1]. Alveg eins og í fyrsta tilvikinu munu vera n-2 samanburðir n sinnum þar sem indexinn mætist ekki í miðjunni fyrr en á seinasta partion-inu.
Þriðja tilvik:
[1,10,2,9,3,8,4,7,5,6]. Gerist svipað í fyrsta og öðru tilviki, þ.e. jaðar tilvikin eða verstu tilvikin eru alltaf partion-inu, eins lítið eða eins stórt og mögulegt er.

Dæmi 4:

```
import java.util.Scanner;

public class quickSort
{

  public static void insertionSort(double[] a, int lo, int hi)
  {
     int N = a.length;
     for (int i = 1; i < N; i++)
     {
        for (int j = i; j > 0 && a[j] < a[j-1]; j--)
           exch(a, j, j-1);
     }
  }
  public static void exch(double [] a, int i, int j)
  {
    double ch = a[i];
    a[i] = a[j];
    a[j] = ch;
  }
  private static int partition(double[] a, int lo, int hi)
  {
    int i = lo, j = hi+1;
    double v = a[lo];
    while (true)
    {
       while (a[++i] < v) if (i == hi) break;
       while (v < a[--j]) if (j == lo) break;
       if (i >= j) break;
       exch(a, i, j);
    }
    exch(a, lo, j);
    return j;
  }
  public static void sort(double[] a, int M)
  {
    StdRandom.shuffle(a);
    sort(a, 0, a.length - 1, M);
  }
  private static void sort(double[] a, int lo, int hi, int M)
  {
     if (hi <= lo) return;
     if((hi - lo) + 1 <= M) insertionSort(a, lo, hi);
     else {
       int j = partition(a, lo, hi);
       sort(a, lo, j-1, M);
       sort(a, j+1, hi, M);
     }
  }

  public static void main(String[] args)
  {
    Scanner sc = new Scanner(System.in);
    int N = sc.nextInt();
    double [] a = new double[N];
    System.out.println("N: " + N);
    for(int i = 0; i < N; i++)
    {
      a[i] = StdRandom.uniform();
    }
    double [] timar = new double[31];
```

```
        double max = 0;
        for(int i = 0; i <= 30; i++)
        {
          double heild = 0;
          for(int j = 0; j < 100; j++)
          {
            long start = System.currentTimeMillis();
            sort(a, i);
            long stop = System.currentTimeMillis();
            heild += (stop - start);
          }
          timar[i] = heild/100;
          if(timar[i] > max)
          {
            max = timar[i];
          }
    }
        StdDraw.setXscale(0, max*1.1);
        StdDraw.setYscale(0, max*1.1);
        StdStats.plotBars(timar);
      }
    }
```





Dæmi 5:

P R I O * R * * I * T * Y * * * Q U E * * * U * E

IOPR
* skilar R,
IOP
IOPR
* skilar R,
IOP
* skilar P
IO
IIO
* skilar O
II
IIT
* skilar T
II
IIY
* skilar Y
* skilar I
* skilar I
EUQ
* skilar Q
* skilar U
* skilar E
U
* skilar U
E

Dæmi 6:
E A S Y Q U E S T I O N

YTUSQNEASIOE