x
Stefan Fischer
Benjamin Neidhardt
Merle Kammer

1	2	3	4	Σ

Übungsblatt Nr. 4

(Abgabetermin 18.05.2017)

Aufgabe 1

a)

,

```
smallest Difference (Array a){
   mergeSort (a);
      int minDiff = a[1] - a[0];
         int res = 0;
              for (int i = 1; i< a.lenght -1; i = i +1){
           int curDiff = a[i+1] - a[i];
         if (curDiff < minDiff)){
      minDiff = curDiff;
   res = i;}
   }
return res;}
```

MergeSort hat bekannterweise eine Laufzeit von $O(nlogn)$ wenn wir jeweils zwei Teilfolgen mischen. Nachdem wir sortiert haben wird jedem Schritt die Differenz zweier Nachbarn bestimmt und mit der aktuellen minimalen Differenz verglichen. Diese Vergleiche benötigen $O(n)$ bis das komplette Array durch ist und somit die minimale Differenz eindeutig bestimmt wurde.
Das Indexpaar ist dann a[res], a[res + 1].
Die Gesamtlaufzeit ist somit:

$$O(nlogn) + O(n) \in O(nlogn)$$

b)

,

```
maxFreq (Array a){
   mergeSort (a); // -> O(nlogn)
      int max = 0;
         int counter = 0;
            int res;
               for (int i = 0; i< a.lenght -1; i = i +1){
                     if (a[i] == a[i+1] ){
                  counter = counter + 1;}
               else {
            if (counter > max){
         max = counter;
      res = a[i];}
      }
   }
return res;}
```

Laufzeit MergSort = $O(nlogn)$

c)

Aufgabe 2

Aufgabe 3

a)

Das Pivotelement ist immer der Median und teilt somit immer genau bei der Hälfte. (Siehe Vorlesung 4 Quicksort: Mittlere Analyse)

b)

Gesucht ist ein Linearzeitalgorithmus: i Häuser mit $1 \leq i \leq n$, Häuser haben $p_i \in N$ Personen
1. erstes und n-tes Haus als initiale Indizes b, e wählen
2. Anzahl Personen be in Haus bestimmen $\rightarrow$ Abstand $d_b + 1 - d_b$ bestimmen $\rightarrow$ Anzahl Personen mal den Abstand $\rightarrow$ ergibt b_s
Anzahl Personen be $\rightarrow$ Abstand $d_e - d_e - 1 \rightarrow$ Anzahl Personen mal Abstand $\rightarrow$ ergibt e_s
3. Wenn $b_s \leq e_s$, dann $p_b + 1 = p_b + p_b + 1$ sonst $p_e - 1 = p_e + p_e - 1$
4. Dann Indexverschiebung: war $b_s < e_s$, dann $b := b + 1, e := e$ sonst $e := e - 1, b := b$
5. Sobald b=e terminiert der Algorithmus

c)

1. Multipliziere niedrigere Personenzahl mit Distanz von A, B. Speicher diesen Wert als Minimum.
2. Falls Summe kleiner als aktuelles Minimum, mache es zu neuem Minimum.
3. Nach Vergleich aller Städte, gib das Minimum zurück

Aufgabe 4

a)

Wahrscheinlichkeit um k mal (hintereinander) Kopf zu werfen ist p^i (analog bei Zahl). Die Wahrscheinlichkeit in festgelegter Reihenfolge beim Werfen k mal Kopf und n-k mal Zahl zu werfen ist also $p^k * (1-p)^n - k$. Die Anzahl an Möglichkeiten, auf die sich die k Erfolge auf n Würfe verteilen können ist: $\binom{n}{k}$. Damit macht die Formel Sinn.

b)

$E(x) = \sum_{x \in M} x \cdot P([X - x)$

$= \sum_{x \in M} x \cdot \binom{n}{x} p^x (1-p)^{n-x}$

$= \sum_{x=0}^{n} x \cdot \binom{n}{x} p^x (1-p)^{n-x} \rightarrow x = 0 fälltaus$

$= \sum_{x=1}^{n} x \cdot \frac{n!}{x!(n-x)} p^x (1-p)^{n-x}$

$= n \cdot \sum_{x=1}^{n} \frac{n!}{x!(n-x)} p^x (1-p)^{n-x}$

$= n \cdot \sum_{x=1}^{n} \frac{(n-1)!}{(x-1)! \cdot (n-x)!} p^x (1-p)^{n-x}$

$= n \cdot p \cdot \sum_{x=1}^{n} \cdot \binom{n-1}{x-1} \cdot p^{x-1} (1-p)^{(n-1)-(x-1)} \rightarrow x' := x - 1$

$= n \cdot p \cdot \sum_{x'=0}^{n-1} \cdot \binom{n-1}{x'} \cdot p^{x'} (1-p)^{(n-1)-x'}$

$= n \cdot p \cdot (p + (1-p))^{n-1} \rightarrow [BinomischerLehrsatz]$

$= \mathrm{n} \cdot p \cdot 1^{n-1} \cdot n \cdot p$

c)

$Var(X) = E(X^2) - E(X)^2$
$= E(X^2) - 2E(X)^2 + E(X)^2$
$= E(X^2) - 2E(X) \cdot E(X) + E(X)^2$
$= E(X^2) - E(2X) \cdot E(X) + E(X)^2$
$= E(X^2) - E(2XE(X)) + E(X)^2$
$= E(X^2 - 2XE(X)) + E(X)^2$
$= E(X^2 - 2XE(X) + E(X)^2)$
$= E((X - E(X))^2)$