Aufgabe 4: Schlüssellöcher

Tobias Nöthlich - Teilnahme-ID: 6396

29.11.2015

Inhaltsverzeichnis

Teil I
Lösungsidee

1. Zuerst galt es die Anzahl der möglichen Lochmuster für zwie verschiedene Produktionsverfahren zu berechnen. Dieser Teil stellt einfach eine Variation mit Zurücklegen dar, weshalb man die Formel n^k verwenden kann. Dabei entspricht n der Anzahl der möglichen Zustände eines Punktes der Karte (Loch oder kein Loch) und k entspricht der Anzahl der Stellen welche ein Loch enthalten können. Dieselbe Formel gilt auch beim zweiten Teil der ersten Teilaufgabe, nur dass diesmal gilt $n^k * n^m$ wobei n denselben Wert wie im ersten Teil annimt, k allerdings für die Stellen links bzw. rechts der mittleren Reihe stehen, m jedoch für die Anzahl der möglichen Löcher in der mittleren Reihe.

2. In der zweiten Teilaufgabe wurde ein Programm gefordert welches für verschiedene Mengen Schlüssel möglichst unterschiedliche Lochmuster ausgibt. Um diese Aufgabe lösen zu können muss als erstes geklärt werden wann sich Schlüssel ähnlich sind und wann nicht. Als ähnlich kann man Schlüssel bezeichnen welche nur an wenigen Stellen Abweichungen im Lochmuster haben. Daraus folgt das man für die Lösung dieser Aufgabe darauf achten muss, dass an möglichst vielen Stellen im Lochmuster Unterschiede auftreten. Umsetzen lässt sich das indem man eine Liste *Zeichen* von einer beliebigen Größe über 2 zufällig mit zwei Zeichen füllt welche für ein Loch bzw. keine Öffnung stehen. Nun kann man aus dieser Liste den ersten Schlüssel auslesen und in ein zweidimensionales Array *Schlüssel* vom Typ char abspeichern. Beim zweiten Schlüssel muss nun auf einen möglichst großen Unterschied geachtet werden,

sodass es nicht mehr ausreicht ihn einfach per Zufall zu generieren. Dafür bietet sich nun ein Abgleich mit dem zuletzt generierten Schlüssel an, um das genaue Gegenteil vom letzten Schlüssel zu speichern. Da allerdings auch dies keine zufriedenstellende Lösung ist (der dritte Schlüssel wäre gleich dem ersten) muss nun auch wieder eine geringe Chance bestehen selbe Zeichen wie beim vorherigen Schlüssel zu setzen. Zusätzlich kann man nach einer bestimmten Anzahl an generierten Schlüsselmustern die Liste mit den Zeichen erneut zufallsgenerieren lassen, was eine noch größere Variation zur Folge hat. Die entstandenen Schlüsselmuster werden direkt nachdem sie in das Array übertragen wurden Zeichen für Zeichen in eine Textdatei geschrieben, da ein Windows-Konsolenfenster bei großen Schlüsselzahlen die ersten Muster nicht mehr anzeigen kann.

Teil II
Umsetzung

1 Teil A

Wie schon in der Lösungsidee erklärt lässt sich die Gesamtzahl der möglichen Schlüssel welche ein Loch in der rechten oberen Ecke und kein Loch in der linken oberen Ecke haben durch 2^{23}bestimmen. Dabei kommt man auf ein Ergebnis von 8.388.608 Möglichkeiten. Der zweite Teil erforderte die Berechnung von $2^{10} * 2^5$. Mit dieser Einschränkung gibt es nur noch 32.768 Möglichkeiten für verschiedene Schlüssel.

2 Teil B

Da ich in der Lösungsidee die wesentlichen Schritte schon genannt habe, bleibt mir in der Umsetzung nicht viel mehr als den Quellcode näher zu erläutern.

2.1 int main()

Die Funktion *int main()* ruft im ersten Schritt die Funktion *void Initialisierung()* auf, welche ihrerseits in einer for-Schleife die Liste *Zeichen* mit Plussen und O's befüllt. Sobald die Liste gefüllt ist, fährt das Programm in der main()-Funktion damit fort, den Benutzer nach der gewünschten Anzahl an Mustern zu fragen. Sobald eine Zahl größer als Null eingegeben wurde, wird eine Textdatei namens "Schlüssel.txt" erstellt - beziehungsweise geöffnet - und in einer weiteren for-Schleife die Funktion *void generiereSchlüssel()* aufgerufen. Ist die Überprüfung für das gesamte Array abgeschlossen, wird in der Funktion *int main()* das fertige Array in die Datei "Schlüssel.txt" geschrieben. Sobald for-Schleife durchgelaufen ist gibt das Programm den Namen der Datei aus, in welche die Schlüssel gespeichert wurden und terminiert mit dem Fehlercode 0.

2.2 void Initialisierung()

In dieser Funktion wird die vorher definierte Liste mit Plussen oder O's befüllt, wobei zufällig gewählt wird wann welches Zeichen in die Liste geschrieben wird.

2.3 void generiereSchlüssel()

Diese Funktion überprüft nun zeilenweise das Array *Schlüssel* und vergleicht es mit der Lise. Sollte das Zeichen im Array mit dem Zeichen, welches als nächstes aus der Liste käme, übereinstimmen, so wird mit einer Chance von 60% das gegenteilige Zeichen gewählt. Desweiteren ruft sie nach jedem 3. Durchlauf dei Funktion *void Initialisierung()* auf, um sicherzustellen dass sich die Muster nicht wiederholen.

Teil III
Beispiele

- Eine Lösung für 7 Schlüssel:

```
O++O+   +OO+O   O+O++   +O+OO   O+O+O
O+OOO   +O+++   ++OOO   OO+++   ++OOO
O+O+O   +++O+   +O++O   OOOO+   ++++O
++OO+   OO++O   ++OO+   OO++O   ++OOO
O++O+   +OO+O   ++OOO   OO++O   +OO++

++OO+   OO++O
OO+O+   ++OOO
O+OOO   +O+++
O++++   +OOOO
OO+O+   ++OO+
```

- Eine Lösung für 20 Schlüssel:

```
OOOO+ +++++ +OOOO O++++ ++OOO +O+++ O+O+O +++O+ O+O+O +O+++ O+OO+ OO++O
O++O+ +OO+O +++O+ OOOOO +++++ OOOOO +++++ OO++O +OO++ O+OOO +O+++ O+OOO
O++O+ +OO+O OOOOO +++++ OOOO+ ++++O O+O++ +O+OO +OOOO O+O++ +O+OO O+OO+
+OO+O O++++ ++O+O OO++O O+OO+ +OO++ OO+OO O++++ OOOOO ++++O OOO++ +++O+
+O+O+ O+++O OOOO+ +++++ OOOO+ O+O+O +O+++ O+OOO +OOO+ O++++ ++OOO O+O+O

+++O+ O+O+O OO+O+ +OO+O O++O+ ++O+O OO+OO +++++
+O+++ O+OO+ OO++O ++OO+ O+++O +OOO+ O+++O +O+O+
++O+O +O+++ O+OO+ O+++O +OOO+ O++++ +OOO+ ++++O
OO+++ ++OO+ ++++O O+OO+ +++++ OOOOO +O+++ ++OO+
+OO++ O++++ +++OO OOO++ +++OO O++++ OOO+O +++O+
```

Wie man sehen kann weisen die Schlüssel große Unterschiede auf.

Teil IV
Quelltext

```
1  #include <iostream>          //Ein-/Ausgabe
2  #include <queue>             // Queue
3  #include <time.h>            //time(NULL)
4  #include <cstdlib>           //system("Pause)
5  #include <fstream>           //in .txt Datei schreiben
6
7
8  #define Pause system("pause")
9  #define QUEUE_SIZE 81
10 #define SIZE_X 5
11 #define SIZE_Y 5
12
13 std::queue<char> Zeichen;
14 char Schlüssel[SIZE_Y][SIZE_X] = { 0 };
15 int Zähler;
16 std::ofstream Schreiben;
17
18 void Initialisierung ()
19 {
20         srand(time(NULL));
21         for (int i = 0; i < QUEUE_SIZE; i++)
22         {
23                 int     Zufallszahl = rand() % 100;
24
25                 if (Zufallszahl<50)
26                 {
27                         Zeichen.push('+');
28                 }
29
30                 if (Zufallszahl>49)
31                 {
32                         Zeichen.push('O');
33                 }
34
35
36         }
37 }
38
39
40 void generiereSchlüssel ()
41 {
```

```
42
43          srand(time(NULL));
44          Zähler++;
45
46          if (Zähler >= 3)
47          {
48                   Initialisierung ();
49                  Zähler = 0;
50          }
51
52          for (int i = 0; i < SIZE_Y; i++)
53          {
54                  for (int j = 0; j < SIZE_X; j++)
55                  {
56                          if (Schlüssel[i][j] == NULL)
57                          {
58                                  Schlüssel[i][j] = Zeichen.front();
59                                  Zeichen.pop();
60                                  Zeichen.push(Schlüssel[i][j]);
61                          }
62
63                          else if (Schlüssel[i][j] == 'O' && Zeichen.front() == 'O')
64                          {
65                                  int Zufallszahl = rand() % 10;
66
67                                  if (Zufallszahl < 6)
68                                  {
69                                          Schlüssel[i][j] = '+';
70                                          Zeichen.pop();
71                                          Zeichen.push(Schlüssel[i][j]);
72                                  }
73
74                                  else
75                                  {
76                                          Schlüssel[i][j] = 'O';
77                                          Zeichen.pop();
78                                          Zeichen.push(Schlüssel[i][j]);
79                                  }
80                          }
81
82                          else if (Schlüssel[i][j] == '+' && Zeichen.front() == '+')
83                          {
84                                  int Zufallszahl = rand() % 10;
85                                  if (Zufallszahl < 6)
86                                  {
87                                          Schlüssel[i][j] = 'O';
```

```
88                                              Zeichen.pop();
89                                              Zeichen.push(Schlüssel[i][j]);
90                                      }
91
92                                      else
93                                      {
94                                              Schlüssel[i][j] = '+';
95                                              Zeichen.pop();
96                                              Zeichen.push(Schlüssel[i][j]);
97                                      }
98
99                              }
100
101                             else
102                             {
103                                     Schlüssel[i][j] = Zeichen.front();
104                                     Zeichen.pop();
105                                     Zeichen.push(Schlüssel[i][j]);
106                             }
107
108                     }
109
110             }
111
112
113     }
114
115     int main()
116     {
117             const unsigned char ue = static_cast<unsigned char>(129); //ü
118             int Schlüsselzahl;
119
120              Initialisierung ();
121
122             std::cout << "Wie viele Schl"<<ue<<"ssel sollen generiert werden?" << std::endl;
123             std::cin >> Schlüsselzahl;
124             while (Schlüsselzahl <= 0)
125             {
126                     std::cout << "Bitte eine Zahl größer Null eingeben!" << std::endl;
127                     std::cin >> Schlüsselzahl;
128             }
129             Schreiben.open("Schlüssel.txt", std::ios::out | std::ios::trunc);
130
131             for (int i = 0;i < Schlüsselzahl; i++)
132             {
133                     generiereSchlüssel ();
```

```
134
135                     for (int i = 0; i < SIZE_Y; i++)
136                     {
137                             for (int j = 0; j < SIZE_X; j++)
138                             {
139                                     Schreiben << Schlüssel[i][j];
140                             }
141                             Schreiben << std::endl;
142                     }
143                     Schreiben << std::endl;
144
145             }
146             Schreiben.close();
147             std::cout << "Die Schl"<<ue<<"ssel wurden in der Datei 'Schl" << ue << "ssel.txt' gespeichert!"
148             Pause;
149             return 0;
150
151     }
```