Radfahrspaß Dokumentation

Bundeswettbewerb Informatik 2016

Johannes Sonn
2016

Inhaltsverzeichnis

Lösungsidee

Anhand eines bestimmten Parcours, welcher in einem Textdokument festgelegt wird soll bestimmt werden, ob es möglich ist diesen Parcours regelkonform befahrbar ist oder nicht. Falls dies möglich ist, soll mithilfe von Plus- und Minuszeichen (in einem Textdokument) ausgegeben werden, auf welchen Geraden der Fahrer beschleunigen(+) und auf welchen er verlangsamen(-) soll. Dabei muss darauf geachtet werden, dass auf jeder Geraden entweder beschleunigt oder verlangsamt werden muss.

Um bestimmen zu können, ob der Parcours regelkonform befahrbar ist muss jeweils die Anzahl der Geraden, der Steigungen und der Gefälle in Variablen (*anzGeraden*, *anzSteigung* , *anzGefaelle*) eingelesen werden.

Zunächst wird die Differenz (*diffSteigGef*) zwischen *anzSteigung* und *anzGefaelle* berechnet. Anschließend wird zwischen drei Fällen unterschieden:

1. *diffSteigGef* > 0
2. *diffSteigGef* < 0
3. *diffSteigGef* = 0

diffSteigGef > 0

Wenn *diffSteigGef* größer als 0 ist, sind mehr Gefälle als Steigungen vorhanden und es wird von *diffSteigGef* die Anzahl der Geraden (*anzGeraden*) abgezogen und in *erg* gespeichert. Dabei wird berechnet ob man die „überschüssige“ Geschwindigkeit abbremsen kann, indem man auf allen Geraden bremst.

erg	Möglich/Unmöglich
erg = 0	**Möglich** ☺ Es kann, wenn man auf allen Geraden bremst die „überschüssige“ Geschwindigkeit ausgeglichen werden, damit man mit 0 m/s im Ziel ankommt.
erg > 0	**Unmöglich** ☹ Es kann, selbst wenn man auf allen Geraden bremst die „überschüssige“ Geschwindigkeit nicht ausgeglichen werden, damit man mit 0 m/s im Ziel ankommt.
erg < 0	**Möglich** ☺ Es kann die „überschüssige“ Geschwindigkeit ausgeglichen werden, damit man mit 0 m/s im Ziel ankommt. Allerdings sind noch Geraden „übrig“. Solange eine gerade Anzahl von Geraden übrig ist, wird auf den letzten Geraden gebremst, damit die überschüssige Energie ausgeglichen wird. Auf den restlichen Geraden wird auf der ersten Hälfte beschleunigt und auf der anderen gebremst. Es wird erst beschleunigt und dann gebremst, damit man nicht während der Fahrt auf dem Parcours stehen bleibt. Wenn eine ungerade Anzahl von Geraden übrig ist, kann der Parcours nicht regelkonform befahren werden. Man würde entweder um 1 m/s zu schnell im Ziel ankommen oder vor dem Ziel stehen bleiben.

diffSteigGef < 0

Wenn *diffSteigGef* kleiner als 0 ist, wird zu *diffSteigGef* die Anzahl der Geraden (*anzGeraden*) addiert und in *erg* gespeichert. Dabei wird berechnet ob man die Geschwindigkeit aufbringen kann, die benötigt wird, um mit 0m/s im Ziel anzukommen, indem man auf allen Geraden beschleunigt.

erg	Möglich/Unmöglich
erg = 0	**Möglich** ☺ Es kann, wenn man auf allen Geraden beschleunigt die benötigte Geschwindigkeit aufgebracht werden, damit man mit 0 m/s im Ziel ankommt.
erg > 0	**Möglich** ☺ Es kann die benötigte Geschwindigkeit aufgebracht werden, damit man mit 0 m/s im Ziel ankommt. Allerdings sind noch Geraden „übrig“. Solange die Anzahl der übrigen Geraden eine gerade Zahl ist, wird auf den ersten Geraden beschleunigt, damit die überschüssige Energie ausgeglichen wird. Auf den restlichen Geraden wird auf der ersten Hälfte beschleunigt und auf der anderen gebremst. Es wird erst beschleunigt und dann gebremst, damit man nicht während der Fahrt auf dem Parcours stehen bleibt. Wenn eine ungerade Anzahl von Geraden übrig ist, kann der Parcours nicht regelkonform befahren werden. Man würde entweder um 1 m/s zu schnell im Ziel ankommen oder vor dem Ziel stehen bleiben.
erg < 0	**Unmöglich** ☹ Es kann, selbst wenn man auf allen Geraden beschleunigt die benötigte Geschwindigkeit nicht aufgebracht werden, damit man mit 0 m/s im Ziel ankommt.

diffSteigGef = 0

Wenn *diffSteigGef* gleich 0 ist, gleichen die Gefälle und Steigungen sich gegenseitig aus. Es wird überprüft, ob die Anzahl der Geraden (*anzGeraden*) eine gerade Zahl ist. Wenn dies so ist, wird auf der ersten Hälfte der Geraden beschleunigt und auf der anderen gebremst, damit sich auch die Geraden gegenseitig ausgleichen. Falls *anzGeraden* eine ungerade Zahl ist, können die Geraden sich nicht gegenseitig ausgleichen und der Parcours ist nicht regelkonform befahrbar.

Umsetzung

Als Hochsprache wird Python verwendet. Das Programm wird über die Kommandozeile ausgeführt. Der Inhalt des angegebenen Textdokuments wird blockweise (je 1024 Zeichen) als String eingelesen. Anschließend werden in diesem String Unterstriche, Backslashes und Slashes gezählt und geprüft ob er mit einem Backslash startet. Danach wird, wie im Abschnitt „Lösungsidee“ beschrieben, geprüft ob der Parcour regelkonform befahrbar ist. Wenn der Parcours nicht regelkonform befahrbar ist wird „Nein“ ausgegeben. Ansonsten wird „Ja“, wie oft beschleunigt und wie oft gebremst werden soll und wo die Loesung gespeichert wurde ausgegeben. In der Lösungs-Textdatei findet man eine Kombination aus Plus- und Minuszeichen, welche dem Benutzer sagen auf welchen Geraden beschleunigt und auf welchen gebremst werden soll.

Das Programm wird nur mit dem Input-Dateipfad und –Namen aufgerufen. Die Lösung, wird in demselben Ort, wie die Input-Datei gespeichert, wobei der Dateiname um "_Loesung.txt" erweitert wird. In der Lösungsdatei stehen Plus- und Minus-Zeichen, welche dem Benutzter sagen sollen auf welchen Geraden beschleunigt (Plus) und auf welchen gebremst (Minus) werden soll. Da manche Parcours sehr lang sind und so sehr große Lösungsdateien entstehen, kann es mit einem Texteditor zu Problemen kommen. Um diese Probleme zu umgehen, kann für das Öffnen der Dateien ein Hex-Editor benutzt werden.

Beispiele

```
~$ python Radfahrspass_024.py parcours0.txt
Nein
~$ python Radfahrspass_024.py parcours1.txt
Ja
Es soll 99847 mal beschleunigt und 33 mal gebremst werden
Die Loesung wurde unter " parcours1_Loesung.txt " gespeichert
~$ python Radfahrspass_024.py parcours2.txt
Ja
Es soll 199973 mal beschleunigt und 188 mal gebremst werden
Die Loesung wurde unter " parcours2_Loesung.txt " gespeichert
~$ python Radfahrspass_024.py parcours3.txt
Ja
Es soll 500825 mal beschleunigt und 321 mal gebremst werden
Die Loesung wurde unter " parcours3_Loesung.txt " gespeichert
~$ python Radfahrspass_024.py parcours4.txt
Ja
Es soll 1001577 mal beschleunigt und 83 mal gebremst werden
Die Loesung wurde unter " parcours4_Loesung.txt " gespeichert
~$ python Radfahrspass_024.py parcours5.txt
Ja
Es soll 1996176 mal beschleunigt und 440 mal gebremst werden
Die Loesung wurde unter " parcours5_Loesung.txt " gespeichert
~$ python Radfahrspass_024.py parcours6.txt
Ja
Es soll 5000833 mal beschleunigt und 393 mal gebremst werden
Die Loesung wurde unter " parcours6_Loesung.txt " gespeichert
~$ python Radfahrspass_024.py parcours7.txt
Ja
Es soll 10002370 mal beschleunigt und 70 mal gebremst werden
Die Loesung wurde unter " parcours7_Loesung.txt " gespeichert
~$ python Radfahrspass_024.py parcours8.txt
Ja
Es soll 19991415 mal beschleunigt und 142 mal gebremst werden
Die Loesung wurde unter " parcours8_Loesung.txt " gespeichert
~$ python Radfahrspass_024.py parcours9.txt
Ja
Es soll 49992957 mal beschleunigt und 252 mal gebremst werden
Die Loesung wurde unter " parcours9_Loesung.txt " gespeichert
~$
```

```
1  # -*- coding: utf-8 -*-
2
3  ############################
4  ### Titel: Radfahrspass ####
5  ### VersionsNr: 25      ###
6  ### Autor: Johannes Sonn##
7  ##########################
8
9  #Import
10 import sys
11 import os
12
13
14 #Variablen werden deklariert
15 diffSteigGef = 0
16 erg = 0
17 anzGeraden = 0
18 anzSteigungen = 0
19 anzGefaelle = 0
20 parcours = ""
21 inputFile = ""
22 outputFile = ""
23 filename = ""
24 outputFilename = ""
25 filesize = 0
26 inputPaket = ""
27 paketSize = 1024
28 anzInputPakete = 0
29 rest = 0
30 i = 0
31 firstChar = ""
32 anzPlus = 0
33 anzMinus = 0
34 USAGE = "USAGE: Radfahrspass [input-file]"
35
36
37 #Falls der Speicherort des "input-files" nicht angegeben ist,
38 #wird ausgegeben wie man dieses Programm richtig aufruft (USAGE)
39 if len(sys.argv) != 2:
40     print USAGE
41
42 #Falls das Programm mit zwei Argumenten aufgerufen wurde...
43 if len(sys.argv) == 2:
44
45     #...wird das Argument, also der Dateipfad in "filename" geschrieben,...
46     filename = str(sys.argv[1])
47
48     #...die Dateigröße in Byte in "filesize" geschrieben,...
49     filesize = os.path.getsize(filename)
50
51     #...die Anzahl der Input-Pakete berechnet und in "anzInputPakete" geschrieben
       und ...
52     anzInputPakete = filesize/paketSize
53
54     #...die restlichen Byte, des Input-Files berechnet und in "rest" geschrieben
55     rest = filesize%paketSize
56
57     #Bei Dateien die kleiner als die bestimmte Paketgröße sind, also
       "anzInputPakete" gleich 0 ...
58     if anzInputPakete == 0: #-- kleine Dateien --
59
60         #...wird die Datei als "inputFile" geöffnet...
61         inputFile = open(filename)
62
63         #...und komplett eingelesen
64         inputPaket = inputFile.read()
65
66
67
```

```
68
69
70              #Danach wird die Anzahl von jeweils Steigungen, Gefällen und Geraden gezählt
71              #und in die entsprechende Variable gespeichert
72              anzSteigungen = inputPaket.count("/")
73              anzGefaelle = inputPaket.count("\\")
74              anzGeraden = inputPaket.count("_")
75
76          #Bei Dateien die größer als die bestimmte Paketgröße sind, also "anzInputPakete"
            größer 0 ...
77          else:
78
79              #...wird die Datei als "inputFile" geöffnet
80              inputFile = open(filename)
81
82              #Für alle Input-Pakete, also solange "i" kleiner als "anzInputPakete" ist
                wird...
83              while i < anzInputPakete:
84
85                  #..."i" um 1 erhöht...
86                  i += 1
87
88                  #...das erste Input-Paket in inputPaket eingelesen...
89                  inputPaket = inputFile.read(paketSize)
90
91                  #... und die Anzahl von jeweils Steigungen, Gefällen und Geraden gezählt
92                  #und in zu der entsprechenden Variable dazu addiert
93                  anzSteigungen += inputPaket.count("/")
94                  anzGefaelle += inputPaket.count("\\")
95                  anzGeraden += inputPaket.count("_")
96
97                  #Falls restliche Zeichen exestieren, also "rest" größer 0 ist...
98                  if rest > 0:
99
100                     #...wird der Rest komplett in "inputPaket" eingelesen...
101                     inputPaket = inputFile.read(rest)
102
103                     #...und anschließend die Anzahl von jeweils Steigungen, Gefällen und
                        Geraden gezählt
104                     #und in zu der entsprechenden Variable dazu addiert
105                     anzSteigungen += inputPaket.count("/")
106                     anzGefaelle += inputPaket.count("\\")
107                     anzGeraden += inputPaket.count("_")
108
109         #Nachdem wird die Datei geschlossen
110         inputFile.close()
111
112         #Es wird die Differenz zwischen der Anzahl der Steigungen (anzSteigungen) und
            der Anzahl der Gefaelle (anzGefaelle) berechnet
113         diffSteigGef = anzGefaelle - anzSteigungen
114
115         #Es wird die Datei nochmal geöffnet, ...
116         inputFile = open(filename)
117
118         #...das erste Zeichen in "firstChar" gespeichert, ...
119         firstChar = inputFile.read(1)
120
121         #...die ParcoursDatei geschlossen
122         inputFile.close()
123
124         #Es wird in "outputFilename" der "filename" mit "_Loesung.txt" als Ergänzung
            geschrieben
125         outputFilename = (str(filename[0:-4]) + "_Loesung.txt")
126
127     #Es überprüft ob der "firstChar" ein Backslash ist, also der Parcours mit einem
        Gefälle beginnt
128     if (firstChar != "\\"):
129         print "ERROR"
130         print "ungueltiger parcours"
```

```
131
132
133
134 #Wenn der Parcours mit einem Gefälle beginnt und diffSteigGef positiv ist,...
135 elif diffSteigGef > 0:
136
137     #...wird die Differenz zwischen "diffSteigGef" und "anzGeraden" in "erg"
        gespeichert, wobei "anzGeraden" der Subtrahend ist
138     erg = diffSteigGef - anzGeraden
139
140     #falls "erg" gleich 0 ist
141     #und somit die 'überschüssige' Geschwindigkeit komplett ausgebremst werden kann,
142     #soll auf jeder Geraden gebremst werden (-)
143     #Dies wird in die Output-Datei geschrieben und es wird ausgegeben wo sich diese
        befindet
144     if erg == 0:
145         print "Ja, der Parcours ist regelkonform befahrbar"
146         anzMinus = anzGeraden
147         outputFile = open(outputFilename, "w")
148         outputFile.write(anzGeraden * "-")
149         outputFile.close()
150         print "Es soll", anzMinus, "mal gebremst werden"
151
152         #Es wird ausgegeben wo die Loesung gespeichert wurde
153         print 'Die Loesung wurde unter "', outputFilename, '" gespeichert'
154
155     #falls "erg" grösser 0 ist
156     #und somit die 'überschüssige' Geschwindigkeit nicht ausgebremst werden kann,
157     #ist der Parcours nicht regelkonform befahrbar (ERROR #1)
158     elif erg > 0:
159         print "Nein, der Parcours ist nicht regelkonform befahrbar"
160
161     #falls "erg" kleiner 0 ist
162     #und somit die 'überschüssige' Geschwindigkeit, durch die Geraden ausgebremst
        werden kann
163     #und noch Geraden 'übrig' sind,
164     #soll auf den letzten Geraden gebremst werden,
165     #damit die 'überschüssige' Geschwindigkeit ausgebremst werden kann
166     #und auf den restlichen Geraden soll auf der ersten Haelfte beschleunigt und auf
        der zweiten gebremst werden.
167     #Dies wird in die Output-Datei geschrieben und es wird ausgegeben wo sich diese
        befindet
168     #
169     #Es wird erst beschleunigt und anschliessend gebremst um zu verhindern,
170     #dass man waehrend der Fahrt auf oder sogar unter eine Geschwindigkeit von 0 m/s
        gelangt
171
172     elif erg < 0:
173         if erg%2 == 0:  #Dies ist aber nur möglich wenn die Anzahl der
            'überschüssigen' Geraden gerade ist
174             print "Ja, der Parcours ist regelkonform befahrbar"
175             anzPlus = (-1*erg) / 2
176             anzMinus = anzGeraden - ((-1*erg) / 2)
177             outputFile = open(outputFilename, "w")
178             outputFile.write(anzPlus * "+" + anzMinus * "-")
179             outputFile.close()
180             print "Es soll", anzPlus, "mal beschleunigt und", anzMinus, "mal
                gebremst werden"
181
182             #Es wird ausgegeben wo die Loesung gespeichert wurde
183             print 'Die Loesung wurde unter "', outputFilename, '" gespeichert'
184
185
186
187 #Wenn diffSteigGef negativ ist,...
188 elif diffSteigGef < 0:
189
190     #...wird die Summe von "diffSteigGef" und "anzGeraden" in "erg" gespeichert
191     erg = diffSteigGef + anzGeraden
```

```
192
193         #falls "erg" gleich 0 ist
194         #und somit die 'benötigte' Geschwindigkeit, um mit 0 m/s im Ziel anzukommen
            komplett 'hergestellt' werden kann,
195         #soll auf jeder Geraden beschleunigt werden (+)
196         #Dies wird in die Output-Datei geschrieben und es wird ausgegeben wo sich diese
            befindet
197         if erg == 0:
198             print "Ja, der Parcours ist regelkonform befahrbar"
199             anzPlus = anzGeraden
200             outputFile = open(outputFilename, "w")
201             outputFile.write(anzGeraden*"+")
202             outputFile.close()
203             print "Es soll", anzPlus, "mal beschleunigt werden"
204
205             #Es wird ausgegeben wo die Loesung gespeichert wurde
206             print 'Die Loesung wurde unter "', outputFilename, '" gespeichert'
207
208         #falls "erg" grösser 0 ist
209         #und somit die 'benötigte' Geschwindigkeit, durch die Geraden 'hergestellt'
            werden kann
210         #und noch Geraden 'übrig' sind,
211         #soll auf den ersten Geraden beschleunigt werden,
212         #damit die 'benötigte' Geschwindigkeit 'hergestellt' werden kann
213         #und auf den restlichen Geraden soll auf der ersten Haelfte beschleunigt und auf
            der zweiten gebremst werden.
214         #Dies wird in die Output-Datei geschrieben und es wird ausgegeben wo sich diese
            befindet
215         #
216         #Es wird erst beschleunigt und anschliessend gebremst um zu verhindern,
217         #dass man waehrend der Fahrt auf oder sogar unter eine Geschwindigkeit von 0 m/s
            gelangt
218         elif erg > 0:
219             if erg%2 == 0:
220                 anzPlus = anzGeraden - (erg) / 2
221                 anzMinus = erg / 2
222                 outputFile = open(outputFilename, "w")
223                 outputFile.write(anzPlus * "+" + anzMinus * "-")
224                 outputFile.close()
225                 print "Ja, der Parcours ist regelkonform befahrbar"
226                 print "Es soll", anzPlus, "mal beschleunigt und", anzMinus, "mal
                    gebremst werden"
227
228                 #Es wird ausgegeben wo die Loesung gespeichert wurde
229                 print 'Die Loesung wurde unter "', outputFilename, '" gespeichert'
230
231         #falls "erg" kleiner 0 ist
232         #und somit die 'benötigte' Geschwindigkeit nicht hergestellt'' werden kann,
233         #ist der Parcours nicht regelkonform befahrbar
234         elif erg < 0:
235             print "Nein, der Parcours ist nicht regelkonform befahrbar"
236
237
238
239     #Wenn diffSteigGef neutral (0) ist...
240     elif diffSteigGef == 0:
241         if anzGeraden%2 == 0:  #...und "anzGeraden" gerade ist,...
242             print "Ja, der Parcours ist regelkonform befahrbar"
243             anzPlus = anzMinus = anzGeraden/2
244             #...soll auf der ersten Haelfte beschleunigt und auf der zweiten gebremst
                werden
245             #Dies wird in die Output-Datei geschrieben und es wird ausgegeben wo sich
                diese befindet
246             outputFile = open(outputFilename, "w")
247             outputFile.write((anzGeraden/2)*"+" + (anzGeraden/2)*"-")
248             outputFile.close()
249             print "Es soll", anzPlus, "mal beschleunigt und", anzMinus, "mal gebremst
                werden"
250
```

```
251             #Es wird erst beschleunigt und anschliessend gebremst um zu verhindern,
252             #dass man waehrend der Fahrt auf oder sogar unter eine Geschwindigkeit von 0
                m/s gelangt
253
254
255
256
257
258
259             #Es wird ausgegeben wo die Loesung gespeichert wurde
260             print 'Die Loesung wurde unter "', outputFilename, '" gespeichert'
261
262         #Sonst ist der Parcours nicht regelkonform befahrbar
263         else:
264             print "Nein, der Parcours ist nicht regelkonform befahrbar"
265
266
267     #Sonst liegt ein Fehler vor
268     else:
269         print "ERROR"
270
```