SEP/Extended SS16

Aus Progwiki
< SEP

Inhaltsverzeichnis

Einleitung

Ziel des erweiterten Beispiels ist es, die bisher erworbenen Programmierkenntnisse zu festigen und das Programm aus dem Basisbeispiel um eine künstliche Intelligenz (K.I.) zu erweitern.

Regeln

Die Funktionalität des Basisbeispiels muss vollständig erhalten bleiben und alle Fehlermeldungen/Allgemeine Hinweise betreffen auch die neuen Befehle.

Neuer Befehl: whoami

Der Befehl bekommt keine Parameter und gibt den Namen eurer K.I. in der folgenden Zeile aus welcher beliebig gewählt werden darf (max. 15 Zeichen, [a-zA-Z0-9], seid kreativ!). Dieser Name erscheint in der Highscore-Liste. Für diesen Befehl muss kein Maze geladen sein.

Beispiel

```
sep> whoami
SuperHero
sep>
```

Neuer Befehl: solve [silent]

Der Befehl solve versucht das Labyrinth vom aktuellen Standpunkt des Spielers aus zu lösen. Ziel ist es einen Pfad ins Ziel zu finden, der die wenigsten Schritte benötigt. (Bonusfelder wirken sich auf die Schrittanzahl natürlich positiv aus.)

Wenn ein Weg ins Ziel gefunden wurde, soll dieser mittels fastmove ausgeführt werden (inklusive Gewinnmeldung). Danach wird ein save-befehl mit dem Parameter '<Name des geladenen Files>Solved ' ('Solved' wird an den

Dateinamen angehängt) ausgeführt. (Bsp: geladenes File: test.maze, Dateiname zum Speichern: test.mazeSolved)

Sollte das Labyrinth lösbar sein wird nach dem Speichern die Zeile

```
The maze was solved in <number of used steps> steps.\nFound path: <path>\n
```

ausgegeben. <number of used steps> ist dabei die Anzahl der Schritte, die der Pfad benötigt hat. <path> ist der gefundene Pfad in Kurznotation, wie bei fastmove. (Achtung: Bonus und QuickSand-Felder zählen zur Schrittanzahl dazu)

Wird der Befehl mit dem optionalen Paramter 'silent' gestartet, wird der Pfad *nicht* ausgegeben, der Befehl wird sonst aber gleich ausgeführt. Die Ausgabe sieht dann folgendermaßen aus:

```
The maze was solved in <number of used steps> steps.\n
```

Fehlermeldungen

- `[ERR] No path found.\n`
  - Wenn vom aktuellen Standpunkt aus kein Weg ins Ziel gefunden werden kann (Bsp: nicht genügend Schritte oder es existiert kein Pfad). In diesem Fall wird auch kein fastmove oder save Befehl ausgeführt.
- `[ERR] You already solved the maze.\n`
  - Wenn sich der Spieler bereits im Ziel befindet.

Allgemeine Hinweise

- Alle testcases für die K.I., die zur Bewertung verwendet werden, werden nur Felder verwenden, die bereits im

Basisbeispiel implementiert wurden. Alle neuen Felder müssen daher nicht berücksichtigt werden. Für die Highscoreliste werden diese Feldtypen aber sehr wohl verwendet.
- Als Algorithmus für die K.I. empfiehlt sich Dijkstra. (Dijkstra kann allerdings nicht mit Bonusfeldern umgehen! Dafür muss man sich selbst einen Ansatz überlegen)
- Der Befehl "solve" darf ein maximales Zeitlimit (Hard-Limit!) von 15 Sekunden nicht überschreiten und nicht mehr als 400 MB Arbeitsspeicher benötigen. Diese Restriktionen sind erforderlich damit wir eure Programme automatisch testen können. Ihr werdet benachrichtigt wenn ihr die Limits überschreitet.

Neuer Befehl: show more nopath

Der Befehl show soll um den optionalen Parameter 'nopath' erweitert werden. Beim Befehl 'show more' soll es möglich sein, den 2. Parameter 'nopath' anzugeben. Dann wird die Zeile, die den zurückgelegten Pfad anzeigt, nicht ausgegeben.

Beispieloutput:

```
sep> show more nopath
Remaining Steps: 32
############################
#+o#Ce##########C<#D   #bEF#
#A+# e#++#++*++#i^##v# ### #
# +####+++++++++>  D#E#     #
# +#B#>++++++#+##############
# A  #B+#++#+++#x   F++++++>#
############################
sep>
```

Die Fehlerbehandlung für den weiteren Parameter soll dabei wie für 'show more' erfolgen.

Neue Feldtypen

Diese 2 Feldtypen kommen beim Aufbaubeispiel neu hinzu:

Counter

```
1-9
```

Das Counter-Feld ist durch die Zahlen 1-9 gekennzeichnet. Die Zahl auf dem Feld zeigt wie oft dieses Feld betreten werden darf. Beim Verlassen des Feldes wird der Counter dekrementiert. Sollte er dabei 0 erreichen, wird das Feld zu einem Wandfeld umgewandelt.

Achtung: Diese Änderungen am Feld dürfen nicht ins File gespeichert werden! (Ein Counterfeld, das bereits zur Wand umgewandelt wurde, wird trotzdem als Counter-Feld mit seinem ursprünglichen Wert in die Datei geschrieben)

Hole

```
s
```

Dieses Feld wird durch den Kleinbuchstaben 's' gekennzeichnet und kann beliebig oft im Labyrinth vorkommen. Wenn der Spieler dieses Feld betritt, wird er wieder auf das Startfeld zurückgebracht.

Die Teleportation selbst verringert die verbleibende Schrittanzahl nicht. Nur das Betreten des Hole-Feldes und das Verlassen des Startfeldes zählen jeweils als 1 Schritt.

Highscore

Als zusätzlichen Anreiz gibt es auch eine Highscore-Liste in der ihr sehen könnt, wie sich euere K.I. im Vergleich zu den anderen Abgaben schlägt:

https://server.nasahl.net/sep/

Alle auf der PALME getätigten Abgaben werden automatisch an die Highscore-Liste weitergeleitet. Dabei werden mit

jeder K.I. mehrere Testcases ausgeführt. Die Anzahl der gelösten Labyrinthe und die dabei benötigten Schritte werden für die Highscore-Liste verwendet.

Die Maps der Highscore findet ihr hier: [letztes Update am 10.06.2016 um 21 Uhr]

http://www.student.tugraz.at/pascal.nasahl/sep/maps.zip

Testcases

Als Hilfestellung für das Aufbaubeispiel findet ihr hier die Testcases, die wir verwendet haben, um das Basic Beispiel zu testen:

https://palme.iicm.tugraz.at/wiki/images/SEP_SS16/testcases.zip [letztes Update am 24.05.2016 um 13 Uhr]

Spezifikation

Einschränkungen

- keine weiteren Ausgaben
- alle Ausgaben müssen auf stdout erfolgen

Erlaubte Bibliotheken

- alle Bibliotheken der C++ Standard Library

Abgabe

- Dateinamen laut Abgabenliste

- Archiv beinhaltet keine Verzeichnisse oder andere Dateien (auch **kein** Makefile)

Abgabenliste

Ab 19.04. bis 9.6.2016 16:00:

- Quellcode (.cpp, .h)

in einem .tar.gz oder .zip Archiv (extended.tar.gz oder extended.zip).

Von „https://palme.iicm.tugraz.at/wiki/SEP/Extended_SS16“