Τεχνολογίες Ευφυών Συστημάτων και Ρομποτική

1η Εργασία (LISP)
Ευθύγραμμο παζλ 6 ψηφίδων

Πανεπιστήμιο Πατρών
Τμήμα Μηχανικών Η/Υ & Πληροφορικής

Μονοπάτης Δημήτριος
ΑΜ: 1040546 (Παλαιός ΑΜ: 4776) - Επί διπλώματι
monopatis@ceid.upatras.gr

28 Απριλίου 2017

Λειτουργικό: GNU/Linux Debian
Χρήση του Lispworks Personal 6.1.1 x86 Linux.
Η αναφορά είναι γραμμένη σε LaTeX .
Η κωδικοποίηση των αρχείων είναι UTF-8 με χρήση για αλλαγή νέων γραμμών του line feed (LF).

I. (α)

Η αναπαράσταση μίας κατάστασης του προβλήματος γίνεται μέσω μίας λίστας εφτά στοιχείων.
Κάθε στοιχείο αντιστοιχεί σε μία θέση, 1η, 2η κλπ.
Έτσι μπορεί να είναι **B** (black-μαύρο) πλακίδιο, **W** (white-λευκό) πλακίδιο ή _ (gap-κενό).

II. (β) Υλοποίηση

Στο αρχείο *pazl.lisp*
(οι κενοί χαρακτήρες μέσα σε string εμφανίζονται ως ␣ μόνο στην παρούσα αναφορά)
Στην αρχή εμφανίζεται μήνυμα στον χρήστη να επιλέξει αν θα εισάγει αυτός αρχική κατάσταση ή θα φορτωθεί ως αρχική αυτή που βρίσκεται στο αρχείο που έχει καθοριστεί στην 4η γραμμή του κώδικα.
Γίνονται έλεγχοι αν οι είσοδοι είναι επιτρεπτές.
Η μετακίνηση γίνεται από την επιθυμητή θέση (μεταξύ 1-7) με την *(move* **θέση***)* πχ (move 2) οπότε θα μετακινηθεί το B ή W δεξιά του _ αν τελευταίο είναι σε θέση μεγαλύτερη του 2ου στοιχείου, αλλιώς αριστερά του.
Η υλοποίηση της μεταφοράς γίνεται στην συνάρτηση *make-move* που ανάλογα την περίπτωση καλεί την subseq καταλλήλως.
Όλες οι κινήσεις καταχωρούνται σε αρχείο που έχει δηλωθεί στην αρχή του κώδικα, καθώς κι εμφανίζονται στην οθόνη.

```
1  (defvar *state*)
2
3  (defvar *game-file-out* "/home/mitsos/lisp/gameout.txt")
4  (defvar *game-file-in* "/home/mitsos/lisp/gamein.txt")
5
6  (defun start ()
7    (display-message-0)
8    (if (= *state* 1)
9      (take-data-1)
10     (if (= *state* 2)
11       (take-data-2)
12     )
13   )
14   (data-check)
15   (format t "H~%~%␣μετακίνηση␣μίας␣ψηφίδας␣γίνεται␣με␣το␣(move␣<position>):␣πχ␣(move␣5)")
16 )
17
18 (defun display-message-0 ()
19   (with-open-file (game-stream-out  *game-file-out* :direction :output
         :if-does-not-exist :create :if-exists :append)
20     (multiple-value-bind
21       (second minute hour date month year day-of-week dst-p tz)
22       (get-decoded-time)
23       (format game-stream-out "~%~d/~2,'0d/~d␣~2,'0d:~2,'0d:~2,'0d"
24        date
25        month
26        year
27        hour
28        minute
29        second
30       ))
31     (format game-stream-out "␣Kalosirthate␣sto␣pazzle␣6␣psifion␣!")
32     (format t "Καλωσήλθατε~%␣στο␣παζλ␣6␣ψηφίδων␣!H~%␣ώρα␣είναι:␣")
33     (multiple-value-bind
34       (second minute hour date month year day-of-week dst-p tz)
35       (get-decoded-time)
36       (format t "~2,'0d:~2,'0d:~2,'0d␣,␣~d/~2,'0d/~d"
37         hour
38         minute
39         second
40         date
41         month
```

```
42          year))
43      (format t "Γράψτε~%␣1␣για␣αρχικοποίηση␣από␣πληκτρολόγιο␣και␣2␣από␣αρχείο:~%")
44      (setf *state* (read))
45    )
46  )
47
48  (defun take-data-1 ()
49    (format t "Δώστε~%␣την␣αρχική␣κατάσταση␣του␣προβλήματος␣με␣την␣μορφή~%")
50    (format t "η~%(<1␣θέση>␣η<2>␣η<3>␣η<4>␣η<5>␣η<6>␣η<7>):(B␣B␣W␣_␣B␣W␣W)")
51    (format t "όπου~%␣B:BlackΜαύρο(),␣W:WhiteΛευκό(),␣_:GapΚενό()")
52    (format t "αυστηρά~%␣3xB,␣3xW␣και␣ένα␣_~%")
53    (format t "Αρχική~%␣κατάσταση:␣")
54    (setf *state* (read))
55    (with-open-file (game-stream-out  *game-file-out* :direction :output
          :if-does-not-exist :create :if-exists :append)
56        (format game-stream-out "~%Arxikh␣katastash:␣~a" *state*)
57    )
58  )
59
60  (defun take-data-2 ()
61    (with-open-file (game-stream-out  *game-file-out* :direction :output
          :if-does-not-exist :create :if-exists :append)
62      (format game-stream-out "~%Read␣from␣file:␣~a~%" *game-file-in*)
63      (format t "Πρέπει~%␣να␣έχετε␣καταχωρήσει␣την␣αρχική␣κατάσταση␣του␣προβλήματος␣με␣την␣μορφή")
64      (format t "η~%(<1␣θέση>␣η<2>␣η<3>␣η<4>␣η<5>␣η<6>␣η<7>)␣πχ:␣(B␣B␣W␣_␣B␣W␣W)")
65      (format t "όπου~%␣B:BlackΜαύρο(),␣W:WhiteΛευκό(),␣_:GapΚενό()")
66      (format t "αυστηρά~%␣3xB,␣3xW␣και␣ένα␣_~%")
67      (format t "Στο~%␣αρχείο:␣~a" *game-file-in*)
68    )
69    (with-open-file (game-stream-in  *game-file-in* :direction :input)
70       (setf *state* (read game-stream-in))
71    )
72    (with-open-file (game-stream-out  *game-file-out* :direction :output
          :if-does-not-exist :create :if-exists :append)
73        (format game-stream-out "Arxikh␣katastash:␣~a" *state*)
74    )
75    (format t "Αρχική~%␣κατάσταση:␣~a~%" *state*)
76  )
77
78  (defun data-check ()
79    (let ((var_b 0)(var_w 0)(var_g 0))
80      (dolist (n *state*)
81        (case n
82          (B (incf var_b 1))
83          (W (incf var_w 1))
84          (_ (incf var_g 1))
85        )
86      )
87      (if (or (/= var_b 3) (/= var_w 3) (/= var_g 1) )
88        (error-1)
89      )
90    )
91  )
92
93  (defun error-1 ()
94    (with-open-file (game-stream-out  *game-file-out* :direction :output
          :if-does-not-exist :create :if-exists :append)
95      (format game-stream-out "~%Error!␣H␣arxikh␣katastash␣den␣einai␣sosti!␣Anagnosi␣apo␣
            to␣pliktrologio.")
96    )
97    (format t "ΣΦΑΛΜΑ~%!␣Η␣αρχική␣κατάσταση␣δεν␣είναι␣σωστή!~%")
98    (take-data-1)
99    (data-check)
100 )
```

```
101
102 (defun make-move (pos)
103   (format t "Μετακίνηση~% από την θέση ~a, νέα κατάσταση:~%" pos)
104   (with-open-file (game-stream-out  *game-file-out* :direction :output
          :if-does-not-exist :create :if-exists :append)
105       (format game-stream-out "~%Metakinhsh apo thn thesh: ~a" pos)
106   )
107   (setq psifida (nth (- pos 1) *state*))
108   (setq gap (position '_ *state*))
109   (if (< (- pos 1) gap)
110     (setf *state* (move-from-left pos psifida gap))
111     (setf *state* (move-from-right pos psifida gap))
112   )
113   (format t "~a" *state*)
114   (with-open-file (game-stream-out  *game-file-out* :direction :output
          :if-does-not-exist :create :if-exists :append)
115       (format game-stream-out ", nea katastash: ~a" *state*)
116   )
117 )
118
119 (defun move-from-left (pos psifida gap)
120   (append
121     (subseq *state* 0 (- pos 1))
122     (subseq *state* pos gap)
123     '(_)
124     (list psifida)
125     (subseq *state* (+ 1 gap))
126   )
127 )
128
129 (defun move-from-right (pos psifida gap)
130   (append
131     (subseq *state* 0 gap)
132     (list psifida)
133     (subseq *state* gap (- pos 1))
134     (subseq *state* pos)
135   )
136 )
137
138 (defun move (pos)
139   (let ((movement (checkpos pos)))
140     (if movement
141         (make-move pos)
142         (display-message-1 pos)
143     )
144   )
145 )
146
147 (defun checkpos (pos)
148   (and (numberp pos) (> pos 0) (< pos 8) (typep pos 'integer) )
149 )
150
151 (defun display-message-1 (pos)
152   (format t "Δεν~% υπάρχει τέτοια θέση: ~a~%" pos)
153   (with-open-file (game-stream-out  *game-file-out* :direction :output
          :if-does-not-exist :create :if-exists :append)
154       (format game-stream-out "~%Den uparxei tetoia thesi: ~a" pos)
155   )
156 )
```

III. (γ) Επίλυση του προβλήματος

Η κατάσταση (B W B _ W B W) είναι η αρχική και η (W W W _ B B B) η τελική ζητούμενη

Με είσοδο:

(start)
1
(B W B _ W B W)
(move 1)
(move 5)
(move 2)
(move 7)

Έχουμε το ζητούμενο αποτέλεσμα. Εμφανίζεται στην οθόνη με ελληνικούς χαρακτήρες και αποθηκεύεται στο αρχείο gameout.txt ως εξής:

19/04/2017 22:06:45 Kalosirthate sto pazzle 6 psifion !
Arxikh katastash: (B W B _ W B W)
Metakinhsh apo thn thesh: 1, nea katastash: (W B _ B W B W)
Metakinhsh apo thn thesh: 5, nea katastash: (W B W _ B B W)
Metakinhsh apo thn thesh: 2, nea katastash: (W W _ B B B W)
Metakinhsh apo thn thesh: 7, nea katastash: (W W W _ B B B)

Στην διαθεσή σας αν συναντήσετε κάποιο πρόβλημα στην εκτέλεση του κώδικα