ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ
ΠΡΟΒΛΗΜΑ ΕΚΚΕΝΩΣΗΣ ΚΤΗΡΙΟΥ

ΟΝΟΜΑΤΕΠΩΝΥΜΟ:
ΠΙΚΡΙΔΑΣ ΜΕΝΕΛΑΟΣ (141291)
ΡΗΓΑΣ ΑΝΔΡΕΑΣ (141257)
ΣΤΑΘΑΚΟΠΟΥΛΟΣ ΚΩΝΣΤΑΝΤΙΝΟΣ (161041)
ΤΜΗΜΑ: ***ΔΕΥΕΤΡΑ 14:00-16:00 (Β2)***

Τμήμα Μηχανικών Πληροφορικής και Υπολογιστών Πανεπιστημίου Δυτικής Αττικής

ΠΕΡΙΕΧΟΜΕΝΑ

Το πρόβλημα της εκκένωσης κτηρίου
Β' (ολοκληρωμένη εκδοχή για το 2ο Εργαστηριακό Μέρος)

ΔΙΑΤΥΠΩΣΗ ΠΡΟΒΛΗΜΑΤΟΣ

Ο κόσμος της εκκένωσης κτηρίου:

Ένα συγκεκριμένο κτήριο διαθέτει ισόγειο χώρο, Ν ορόφους και ένα ασανσέρ για την μετακίνηση των ενοίκων από όροφο σε όροφο. Σε κάθε όροφο θεωρείται γνωστό τα πλήθη των ενηλικών και των ανήλικων ατόμων του που κατοικούν σε αυτόν και τα οποία θα πρέπει να μετακινηθούν στο ισόγειο.

Όταν έρχεται το σήμα εκκένωσης από την αστυνομία, με γνωστά όλα τα παραπάνω στοιχειά, το ασανσέρ μπορεί να βρίσκεται εν αναμονή (άδειο) σε οποιονδήποτε από

τους ορόφους. Η χωρητικότητα του ασανσέρ σε άτομα δεν είναι γνωστή στην αστυνομία αλλά ζητείται κατά την αρχικοποίηση της όλης διαδικασίας. Οι δυνατές ενέργειες είναι:

- Το ασανσέρ μπορεί να κινηθεί από οποιονδήποτε όροφο προς οποιονδήποτε όροφο αρκεί να μην είναι πλήρες και επιπλέον στον όροφο να περιμένει τουλάχιστον ένας ενοϊκός.
- Όταν το ασανσέρ φτάνει σε έναν όροφο, ανοίγει η πόρτα και επιτρέπεται να επιβιβαστούν τόσοι ένοικο όσοι χωρούν και οι υπόλοιποι (αν υπάρχουν) περιμένουν να έρθει το ασανσέρ εκ νέου. Αν υπάρχουν ανήλικοι εν αναμονή, έχουν προτεραιότητα στην επιβίβαση ως εξής:

  α) Αν το ασανσέρ είναι άδειο και υπάρχουν ενήλικες και ανήλικοι στον όροφο, επιβιβάζεται ένας μονό ενήλικας και όσοι ανήλικοι χωρούν, **β)** αν στο ασανσέρ βρίσκεται ήδη τουλάχιστον ένας ενήλικας από άλλον όροφο, τότε επιβιβάζονται όσοι ανήλικοι χωρούν, **γ)** αν το ασανσέρ είναι άδειο και στον όροφο υπάρχουν μονό ανήλικοι τότε επιβιβάζονται όσοι χωρούν.
- Το ασανσέρ κινείται από οποιονδήποτε όροφο προς το ισόγειο μονό στην περίπτωση που είναι πλήρες ή που δεν είναι μεν πλήρες αλλά δεν περιμένει άλλος ένοικος σε κάποιον όροφο προς επιβίβαση.

ΜΟΝΤΕΛΟΠΟΙΗΣΗ ΠΡΟΒΛΗΜΑΤΟΣ

Χώρος καταστάσεων

Οι επιτρεπτές καταστάσεις του προβλήματος αποτελούνται από το συνδυασμό των σχέσεων μεταξύ των αντικειμένων του. Μια κατάσταση ορίζεται από τους ορόφους του κτηρίου (ισόγειο,1ος-Νος όροφος), από το πλήθος των ενοίκων (ενήλικες, ανήλικοι) σε κάθε όροφο και από το πλήθος ενοίκων που χωράνε στο ασανσέρ.

Αρχική κατάσταση

Το ασανσέρ βρίσκεται σε αναμονή σε έναν οποιοδήποτε όροφο και είναι άδειο.

Τελική κατάσταση - Στόχος

Το κτήριο είναι άδειο.

Κανόνες

Σε αυτό το απλό περιβάλλον, σε κάθε κατάσταση μπορούν να εφαρμοστούν οι εξής κανόνες:

- Ο κανόνας init (με την μεγαλύτερη προτεραιότητα) που επιλέγει ως αρχική λειτουργία την είσοδο των δεδομένων του κτηρίου που γίνεται εκκένωση.
- Ο κανόνας end_of_evacuation που προσθέτει το fact της τελικής κατάστασης και εμφανίζει κατάλληλο μήνυμα.
- Ο κανόνας go_to_0_A που το ασανσέρ επιστρέφει στο ισόγειο όταν έχει γεμίσει.
- Ο κανόνας go_to_base_B που ελέγχει αν υπάρχουν ένοικοι στους ορόφους και εμφανίζει κατάλληλο μήνυμα.
- Ο κανόνας move_to_any που μετακινεί το ασανσέρ σε οποιοδήποτε όροφο.
- Ο κανόνας enter_from_any που κάνει εισαγωγή των ενοίκων στο ασανσέρ από οποιοδήποτε όροφο.

Τρόπος Αναπαράστασης του Κόσμου του Προβλήματος

Κωδικοποίηση των κανόνων

➔ *Κανόνας ini*

```
(deffacts ini       "initial facts"
      (capacity 5); SHOWS THE MAXIMUM CAPACITY OF THE ELEVATOR
    (elevator is_at 0 has 5); SHOWS IN WHICH FLOOR THE ELEVATOR IS AND HOW MANY PEOPLE
HAS INSIDE
    (floor 1 has 4 and 2); SHOWS HOW MANY PEOPLE (CHILDERN AND ADULTS) FLOOR 1 HAS
    (floor 2 has 4 and 2); SHOWS HOW MANY PEOPLE (CHILDERN AND ADULTS) FLOOR 2 HAS
    (floor 3 has 4 and 2); SHOWS HOW MANY PEOPLE (CHILDERN AND ADULTS) FLOOR 3 HAS
)
```

σκοπός του κανόνα αυτού είναι να αρχικοποιηθούν τα δεδομένα του προβλήματος. Ο χρήστης μπορεί να αλλάξει τα δεδομένα

➢ *Κανόνας end_of_evacuation*

```
(defrule end_of_evacuation        "building evacuated"
(elevator is_at 0 has 0)
(floor ?floor has ?p and ?c)
      (floor ?floor2&~?floor has ?p&0 and ?c&0)
      =>
       (printout t "END OF EVACUATION" crlf)
       (halt)
)
```

Ο σκοπός αυτού του κανόνα είναι να δείξει ότι το κτήριο έχει αδειάσει εντελώς.

- *Κανόνας go_to_0_A*

```
(defrule go_to_0_A      "elevator is full"
(declare (salience 10))
?i<- (elevator is_at ?x has 5)
(floor ?floor has ?p&~0 and ?c&~0)
      =>
    (retract ?i)
    (assert (elevator is_at 0 has 0))
)
```

Ο σκοπός αυτού του κανόνα είναι να επιστρέφει το ασανσέρ στο ισόγειο όταν είναι γεμάτο.

- *Κανόνας go_to_base_B*

```
?i<- (elevator is_at ?x has ?y)
(floor ?floor&~?x has ?p&0 and ?c&0)
=>
 (if (> ?y 0)
            then
            (retract ?i)
            (assert (elevator is_at 0 has 0))
            (printout t "ALL FLOORS ARE CLEAR" crlf)
      )
)
```

Ο σκοπός αυτού του κανόνα είναι όταν έχουν αδειάσει όλοι οι όροφοι να εμφανίζει αντίστοιχο μήνυμα και να επιστρέφει το ασανσέρ στο ισόγειο.

- *Κανόνας move_to_any*

```
(defrule move_to_any          "elevator moves to any floor"
?i <- (elevator is_at floor ?x has ?y)
   (floor ?floor2&~?x has ?p and ?c)
 =>
(if (> ?p 0)
      then
      (retract ?i)
      (assert (elevator is_at floor ?x has ?y))
)
)
```

Ο σκοπός αυτού του κανόνα είναι να μετακινεί το ασανσέρ από όροφο σε όροφο με βάση τα άτομα που έχει (αν έχει) και τα άτομα του κάθε ορόφου.

- *Κανόνας enter_from_any*

```
;RULE FOR PEOPLE ENTERING THE ELEVATOR FROM ANY FLOOR
(defrule enter_from_any      "people from any floor enter the elevator"
      (declare (salience 5))
 ?i<- (elevator is_at ?x has ?y)
 ?j<- (floor ?floor2 has ?p and ?c)
      (capacity ?cap)
   =>
      (if (and (neq ?x ?floor2) (> ?p 0) (> ?c 0));CASE WHERE THERE ARE ADULTS AND CHILDREN IN THE FLOOR
     then
        (if (and (> ?p 0) (> ?c 0) (>= (- ?cap (+ ?p ?y)) 0) ); CASE WHERE EVERYONE CAN FIT IN THE ELEVATOR
                  then
                (retract ?i ?j)
                  (assert (floor ?floor2 has 0 and 0))
            (assert (elevator is_at ?floor2 has (+ ?p ?y)))

       else (if (and (neq ?x ?floor2) (> ?c 0) (>= (- ?cap (+ ?y ?c 1)) 0));CASE WHERE ALL CHILDREN PLUS ONE ADULT CAN FIT IN THE ELEVATOR
                  then
```

```
                (retract ?i ?j)
                  (assert (floor ?floor2 has (- ?p ?c 1) and 0))
             (assert (elevator is_at ?floor2 has (+ ?y ?c 1)))
                  )


             else (if (and (neq ?x ?floor2) (> ?c 0) (< (- (- ?cap ?y 1) ?c) 0));CASE
WHERE ALL CHILDREN PLUS ONE ADULT CAN NOT FIT IN THE ELEVATOR
                  then
                  (retract ?i ?j)
                  (assert (floor ?floor2 has (- ?p (- ?c (- ?cap ?y 1))) and (- ?c (-
?cap ?y 1))))
                  (assert (elevator is_at ?floor2 has ?cap))
                  )
             )
    else (if (and (neq ?x ?floor2) (> ?p 0) (= ?c 0));CASE IF THERE PEOPLE BUT NO CHILDREN
              then
             (if (and (neq ?x ?floor2) (= ?c 0) (>= (- ?cap (+ ?p ?y)) 0)); CASE WHERE
ALL ADULTS CAN FIT IN THE ELEVATOR
                  then
                (retract ?i ?j)
                  (assert (floor ?floor2 has 0 and 0))
             (assert (elevator is_at ?floor2 has (+ ?y ?p)))
             else (if (and (neq ?x ?floor2) (= ?c 0) (< (- ?cap (+ ?p ?y)) 0)); CASE
WHERE ALL ADULTS CAN NOT FIT IN THE ELEVATOR
                  then
                  (retract ?i ?j)
                  (assert (floor ?floor2 has (- ?p (- ?cap ?y)) and 0))
                  (assert (elevator is_at ?floor2 has ?cap))
             )
        )
      )
  )
)
```

Ο σκοπός αυτού του κανόνα είναι να εισάγει στο ασανσέρ τους ενοίκους με βάση το διαθέσιμο χώρο και το αν είναι ενήλικοι ή ανήλικοι.

Αποτελέσματα εκτέλεσης

1° ΕΝΔΕΙΚΤΙΚΟ ΤΡΕΞΙΜΟ

```
CLIPS> (load "C:/Users/Admin/Desktop/FINAL CODE.CLP")
Defining deffacts: ini
Defining defrule: end_of_evacuation +j+j+j+j
Defining defrule: go_to_0_A +j+j+j
Defining defrule: go_to_base_B +j+j+j
Defining defrule: move_to_any +j+j+j
Defining defrule: enter_from_any =j+j+j+j
TRUE
CLIPS> (reset)
<== f-0     (initial-fact)
==> f-0     (initial-fact)
==> f-1     (capacity 5)
==> f-2     (elevator is_at 0 has 5)
==> f-3     (floor 1 has 4 and 2)
==> f-4     (floor 2 has 4 and 2)
==> f-5     (floor 3 has 4 and 2)
CLIPS> (run 1)
FIRE    1 go_to_0_A: f-2,f-5
<== f-2     (elevator is_at 0 has 5)
==> f-6     (elevator is_at 0 has 0)
CLIPS> (run 1)
FIRE    1 enter_from_any: f-6,f-5,f-1
<== f-6     (elevator is_at 0 has 0)
<== f-5     (floor 3 has 4 and 2)
==> f-7     (floor 3 has 0 and 0)
==> f-8     (elevator is_at 3 has 4)
CLIPS> (run 1)
FIRE    1 enter_from_any: f-8,f-7,f-1
CLIPS> (run 1)
FIRE    1 enter_from_any: f-8,f-4,f-1
<== f-8     (elevator is_at 3 has 4)
<== f-4     (floor 2 has 4 and 2)
```

```
==> f-9     (floor 2 has 3 and 2)
==> f-10    (elevator is_at 2 has 5)
CLIPS> (run 1)
FIRE    1 go_to_0_A: f-10,f-9
<== f-10    (elevator is_at 2 has 5)
==> f-11    (elevator is_at 0 has 0)
CLIPS> (run 1)
FIRE    1 enter_from_any: f-11,f-9,f-1
<== f-11    (elevator is_at 0 has 0)
<== f-9     (floor 2 has 3 and 2)
==> f-12    (floor 2 has 0 and 0)
==> f-13    (elevator is_at 2 has 3)
CLIPS> (run 1)
FIRE    1 enter_from_any: f-13,f-12,f-1
CLIPS> (run 1)
FIRE    1 enter_from_any: f-13,f-7,f-1
CLIPS> (run 1)
FIRE    1 enter_from_any: f-13,f-3,f-1
<== f-13    (elevator is_at 2 has 3)
<== f-3     (floor 1 has 4 and 2)
==> f-14    (floor 1 has 4 and 1)
==> f-15    (elevator is_at 1 has 5)
CLIPS> (run 1)
FIRE    1 go_to_0_A: f-15,f-14
<== f-15    (elevator is_at 1 has 5)
==> f-16    (elevator is_at 0 has 0)
CLIPS> (run 1)
FIRE    1 enter_from_any: f-16,f-14,f-1
<== f-16    (elevator is_at 0 has 0)
<== f-14    (floor 1 has 4 and 1)
==> f-17    (floor 1 has 0 and 0)
==> f-18    (elevator is_at 1 has 4)
CLIPS> (run 1)
FIRE    1 enter_from_any: f-18,f-17,f-1
CLIPS> (run 1)
FIRE    1 enter_from_any: f-18,f-12,f-1
CLIPS> (run 1)
FIRE    1 enter_from_any: f-18,f-7,f-1
```

```
CLIPS> (run 1)
FIRE    1 go_to_base_B: f-18,f-12
<== f-18    (elevator is_at 1 has 4)
==> f-19    (elevator is_at 0 has 0)
ALL FLOORS ARE CLEAR
CLIPS> (run 1)
FIRE    1 enter_from_any: f-19,f-17,f-1
CLIPS> (run 1)
FIRE    1 enter_from_any: f-19,f-12,f-1
CLIPS> (run 1)
FIRE    1 enter_from_any: f-19,f-7,f-1
CLIPS> (run 1)
FIRE    1 end_of_evacuation: f-19,f-17,f-12
END OF EVACUATION
[PRCCODE4] Execution halted during the actions of defrule end_of_evacuation.
```

2ο ΕΝΔΕΙΚΤΙΚΟ ΤΡΕΞΙΜΟ

```
==> f-0     (initial-fact)
CLIPS> (load "C:/Users/Admin/Desktop/FINAL CODE.CLP")
Defining deffacts: ini
Defining defrule: end_of_evacuation +j+j+j+j
Defining defrule: go_to_0_A +j+j+j
Defining defrule: go_to_base_B +j+j+j
Defining defrule: move_to_any +j+j+j
Defining defrule: enter_from_any =j+j+j+j
TRUE
CLIPS> (reset)
<== f-0     (initial-fact)
==> f-0     (initial-fact)
==> f-1     (capacity 4)
==> f-2     (elevator is_at 0 has 0)
==> f-3     (floor 1 has 5 and 1)
==> f-4     (floor 2 has 6 and 3)
==> f-5     (floor 3 has 4 and 0)
CLIPS> (run 1)
FIRE    1 enter_from_any: f-2,f-5,f-1
<== f-2     (elevator is_at 0 has 0)
```

```
<== f-5     (floor 3 has 4 and 0)
==> f-6     (floor 3 has 0 and 0)
==> f-7     (elevator is_at 3 has 4)
CLIPS> (run 1)
FIRE    1 go_to_0_A: f-7,f-4
<== f-7     (elevator is_at 3 has 4)
==> f-8     (elevator is_at 0 has 0)
CLIPS> (run 1)
FIRE    1 go_to_base_B: f-8,f-6
CLIPS> (run 1)
FIRE    1 enter_from_any: f-8,f-6,f-1
CLIPS> (run 1)
FIRE    1 enter_from_any: f-8,f-4,f-1
<== f-8     (elevator is_at 0 has 0)
<== f-4     (floor 2 has 6 and 3)
==> f-9     (floor 2 has 2 and 0)
==> f-10    (elevator is_at 2 has 4)
CLIPS> (run 1)
FIRE    1 go_to_0_A: f-10,f-3
<== f-10    (elevator is_at 2 has 4)
==> f-11    (elevator is_at 0 has 0)
CLIPS> (run 1)
FIRE    1 go_to_base_B: f-11,f-6
CLIPS> (run 1)
FIRE    1 enter_from_any: f-11,f-9,f-1
<== f-11    (elevator is_at 0 has 0)
<== f-9     (floor 2 has 2 and 0)
==> f-12    (floor 2 has 0 and 0)
==> f-13    (elevator is_at 2 has 2)
CLIPS> (run 1)
FIRE    1 go_to_base_B: f-13,f-6
<== f-13    (elevator is_at 2 has 2)
==> f-14    (elevator is_at 0 has 0)
ALL FLOORS ARE CLEAR
CLIPS> (run 1)
FIRE    1 end_of_evacuation: f-14,f-12,f-6
END OF EVACUATION
[PRCCODE4] Execution halted during the actions of defrule end_of_evacuation.
```

Αρχικά θέλαμε να υλοποιήσουμε την εργασία χρησιμοποιώντας templates και έναν πιο φιλικό προς τον χρήστη τρόπο, όμως ο κώδικας μας δεν μας έδειχνε τα αναμενόμενα αποτελέσματα. Ως εκ τούτου ο κώδικας μας υλοποιήθηκε χωρίς templates, με δήλωση της αρχικής κατάστασης εντός του κανόνα "ini".

Για την ορθή λειτουργία του προγράμματος, αν επιθυμείτε να αλλάξετε την μέγιστη χωρητικότητα του ασανσέρ, έκτος από τον κανόνα "ini" η χωρητικότητα θα πρέπει κάθε φορά να αλλάζεται και εντός του κανόνα "go_to_0_A".Τέλος να διευκρινίσουμε ότι:

(floor 1 has 4 and 2). Ο αριθμός 4 υποδηλώνει τα συνολικά άτομα που έχει ο όροφος, δηλαδή ενήλικες και παιδιά μαζί, ενώ ο αριθμός 2 υποδηλώνει πόσα παιδιά υπάρχουν στον όροφο.

Σας επισυνάπτουμε και τα δύο αρχεία κώδικα μαζί με τον κώδικα που ζητούσατε για το μέρος Β της εργασίας. Το μέρος Β του κώδικα είχε παραδοθεί εντός προθεσμίας.