ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ

ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ
UNIVERSITY OF WEST ATTICA

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

ΕΦΑΡΜΟΓΗ ΑΛΓΟΡΙΘΜΩΝ ΑΝΑΖΗΤΗΣΗΣ ΣΤΟ ΠΡΟΒΛΗΜΑ ΤΟΥ PACMAN

ΣΤΟΙΧΕΙΑ ΦΟΙΤΗΤΗ

ΟΝΟΜΑΤΕΠΩΝΥΜΟ : ΑΘΑΝΑΣΙΟΥ ΒΑΣΙΛΕΙΟΣ ΕΥΑΓΓΕΛΟΣ
ΑΡΙΘΜΟΣ ΜΗΤΡΩΟΥ : 19390005
ΕΞΑΜΗΝΟ ΦΟΙΤΗΤΗ : 7ο
ΚΑΤΑΣΤΑΣΗ ΦΟΙΤΗΤΗ : ΠΡΟΠΤΥΧΙΑΚΟ
ΠΡΟΓΡΑΜΜΑ ΣΠΟΥΔΩΝ : ΠΑΔΑ
ΚΑΘΗΓΗΤΗΣ ΕΡΓΑΣΤΗΡΙΟΥ : ΜΑΣΤΟΡΟΚΩΣΤΑΣ ΠΑΡΙΣ
ΚΑΘΗΓΗΤΡΙΑ ΕΡΓΑΣΤΗΡΙΟΥ : ΤΣΕΛΕΝΤΗ ΠΑΝΑΓΙΩΤΑ
ΗΜΕΡΟΜΗΝΙΑ ΠΑΡΑΔΟΣΗΣ : 10/1/2023

ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ

ΦΩΤΟΓΡΑΦΙΑ ΦΟΙΤΗΤΗ :

ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ

1. Ο κόσμος του προβλήματος

Ο κόσμος του pacman περιλαμβάνει τα εξής:

Αντικείμενα : Ο pacman, τα φρούτα, τα κελιά, το πλέγμα των κελιών και το πλέγμα των πλεγμάτων των κελιών

Ιδιότητες : Ο pacman μπορεί να μετακινηθεί δεξιά, αριστερά, πάνω, κάτω ή και να φάει φρούτο. Τα φρούτα συνυπάρχουν σε κελιά όσο δεν βρίσκονται στο ίδιο κελί με τον pacman.

Σχέσεις : Τόσο τα φρούτα όσο και ο pacman βρίσκονται σε κελιά και το κάθε κελί ανήκει σε πλέγμα. Το κάθε πλέγμα ανήκει σε ένα άλλο μεγαλύτερο πλέγμα. Ο pacman μπορεί να φάει φρούτο μόνο στην περίπτωση που βρίσκεται σε κελί που περιέχει φρούτο. Ένα φρούτο μπορεί να περιλαμβάνεται σε ένα κελί μαζί με τον pacman ή να είναι μόνο του.

1.1 Η αρχική και τελική κατάσταση του προβλήματος

Αρχική κατάσταση του προβλήματος μπορεί να οριστεί οποιαδήποτε κατάσταση με την οποία ο pacman και τα φρούτα βρίσκονται σε ένα κελί εντός του πλέγματος που περιλαμβάνει πλέγματα με κελιά. Για παράδειγμα σε πλέγμα με διαστάσεις 4x4 αρχική κατάσταση μπορεί να οριστεί ως εξής :

Πίνακας 1.1.1 Η αρχική κατάσταση

	0	**1**	**2**	**3**
0	[[[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘ ’]	[‘ ‘, ‘f’]]
1	[[‘ ‘, ‘ ’]	[‘ ‘, ‘ ‘]	[‘p‘, ‘ ’]	[‘ ‘, ‘f’]]
2	[[‘ ‘, ‘ ‘]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘ ’]]
3	[[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]]]

Τελική κατάσταση του προβλήματος μπορεί να είναι οποιαδήποτε κατάσταση με την οποία ο pacman έχει φάει όλα τα φρούτα, δηλαδή, βρίσκεται μόνος του σε ένα κελί του πλέγματος. Για παράδειγμα σε πλέγμα με διαστάσεις 4x4 τελική κατάσταση μπορεί να οριστεί ως εξής:

Πίνακας 1.1.2 Η τελική κατάσταση

	0	**1**	**2**	**3**
0	[[[‘ ‘, ‘ ’]	[‘ ‘, ‘ ’]	[‘ ‘, ‘ ’]	[‘ ‘, ‘ ’]]
1	[[‘ ‘, ‘ ’]	[‘ ‘, ‘ ‘]	[‘ ‘, ‘ ’]	[‘ ‘, ‘ ’]]
2	[[‘ ‘, ‘ ‘]	[‘ ‘, ‘ ’]	[‘ ‘, ‘ ’]	[‘ ‘, ‘ ’]]
3	[[‘ ‘, ‘ ’]	[‘ ‘, ‘ ’]	[‘ ‘, ‘ ’]	[‘p‘, ‘ ’]]]

Ο κόσμος του προβλήματος με βάση το στιγμιότυπο του Πίνακα 1 μπορεί να αναπαρασταθεί ως εξής:

Πίνακας 1.1.3 Ο κόσμος του προβλήματος

Αντικείμενα	Ιδιότητες	Σχέσεις
Pacman	Μπορεί να μετακινηθεί στο δεξί ή στο αριστερό ή στο πάνω ή στο κάτω κελί.	Ο pacman βρίσκεται στο κελί του 2ου πλέγματος κελιών, δηλαδή στην θέση (1, 2).
Φρούτο	Δεν μπορεί να μετακινηθεί παρά μόνο να φαγωθεί από τον pacman.	Δέκα φρούτα υπάρχουν ελεύθερα στα κελιά (0, 0), (3, 0), (0, 1), (2, 1), (3, 1), (2, 2), (3, 2), (0, 3), (1, 3) και (3, 3).
Κελί	Το κελί παρέχει δύο θέσεις για το φρούτο και τον pacman.	Το κελί μπορεί να περιλαμβάνει φρούτο ή τον pacman ή φρούτο μαζί με τον pacman ή και τίποτα.
Πλέγμα κελιών	Το πλέγμα κελιών παρέχει τέσσερα κελιά.	Το πλέγμα κελιών περιλαμβάνει τα κελιά.
Πλέγμα πλεγμάτων κελιών	Το πλέγμα πλεγμάτων κελιών παρέχει τέσσερα πλέγματα που περιλαμβάνουν κελιά.	Το πλέγμα πλεγμάτων κελιών περιλαμβάνει τα πλέγματα κελιών.

2. Οι τελεστές μετάβασης

Οι τελεστές μετάβασης είναι αυτοί που επιτρέπουν την χρήση ιδιοτήτων του pacman μέσα στο πλέγμα και που δημιουργούν νέες καταστάσεις του προβλήματος. Συνολικά, στο πρόβλημα του pacman υπάρχουν 5 τελεστές, αυτός της μετακίνησης προς τα δεξιά (move_right), της μετακίνησης προς τα αριστερά (move_left), της μετακίνησης προς τα πάνω (move_up), της μετακίνησης προς τα κάτω (move_down) και του φαγώματος φρούτου (eat). Ο κάθε τελεστής εκτελείται με βάση κάποιες προϋποθέσεις που παρουσιάζονται αναλυτικά στον παρακάτω πίνακα. Οι προϋποθέσεις αυτές ελέγχονται από τις αντίστοιχες συναρτήσεις can_x(όπου x ο αντίστοιχος τελεστής).

Πίνακας 2.1 Οι τελεστή μετάβασης

Τελεστής	Χρήστης	Ιδιότητα	Προϋπόθεση
move_right	Pacman	Ο pacman κινείται σε κελί που βρίσκεται στα δεξιά του.	Ο pacman να μην βρίσκεται στο τελευταίο κελί ενός πλέγματος κελιών.
move_left	Pacman	Ο pacman κινείται σε κελί που βρίσκεται στα αριστερά του.	Ο pacman να μην βρίσκεται στο πρώτο κελί ενός πλέγματος κελιών.
move_up	Pacman	Ο pacman κινείται σε κελί που βρίσκεται από πάνω του.	Ο pacman να μην βρίσκεται στο πρώτο πλέγμα κελιών.
move_down	Pacman	Ο pacman κινείται σε κελί που βρίσκεται από κάτω του.	Ο pacman να μην βρίσκεται στο τελευταίο πλέγμα κελιών.
eat	Pacman	Ο pacman τρώει φρούτο.	Ο pacman να βρίσκεται στο ίδιο κελί με φρούτο.

3. Ο χώρος καταστάσεων

Ο χώρος καταστάσεων του προβλήματος είναι κάθε επιτρεπτή κατάσταση που δημιουργείται με την παρέμβαση των τελεστών μετάβασης. Επιτρεπτή κατάσταση είναι αυτή που ένα τουλάχιστον κελί περιλαμβάνει έναν και μοναδικό pacman. Για παράδειγμα :

Πίνακας 3.1 Η αρχική κατάσταση

	0	**1**	**2**	**3**
0	[[[‘ ‘, ‘f’]]	[‘ ‘, ‘f’]	[‘ ‘, ‘ ’]	[‘ ‘, ‘f’]]
1	[[‘ ‘, ‘ ’]]	[‘ ‘, ‘ ‘]	[‘p‘, ‘ ’]	[‘ ‘, ‘f’]]
2	[[‘ ‘, ‘ ‘]]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘ ’]]
3	[[‘ ‘, ‘f’]]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]]]

Πίνακας 3.2 Η κατάσταση μετά την παρέμβαση του τελεστή move_left

	0	**1**	**2**	**3**
0	[[[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘ ’]	[‘ ‘, ‘f’]]
1	[[‘ ‘, ‘ ’]	[‘p‘, ‘ ‘]	[‘ ‘, ‘ ’]	[‘ ‘, ‘f’]]
2	[[‘ ‘, ‘ ‘]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘ ’]]
3	[[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]]]

Πίνακας 3.3 Η κατάσταση μετά την παρέμβαση του τελεστή move_down

	0	**1**	**2**	**3**
0	[[[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘ ’]	[‘ ‘, ‘f’]]
1	[[‘ ‘, ‘ ’]	[‘ ‘, ‘ ‘]	[‘ ‘, ‘ ’]	[‘ ‘, ‘f’]]
2	[[‘ ‘, ‘ ‘]	[‘p‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘ ’]]
3	[[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]]]

Πίνακας 3.4 Η κατάσταση μετά την παρέμβαση του τελεστή eat

	0	**1**	**2**	**3**
0	[[[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘ ’]	[‘ ‘, ‘f’]]
1	[[‘ ‘, ‘ ’]	[‘ ‘, ‘ ‘]	[‘ ‘, ‘ ’]	[‘ ‘, ‘f’]]
2	[[‘ ‘, ‘ ‘]	[‘p‘, ‘ ’]	[‘ ‘, ‘f’]	[‘ ‘, ‘ ’]]
3	[[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]]]

Πίνακας 3.5 Η κατάσταση μετά την παρέμβαση του τελεστή move_right

	0	**1**	**2**	**3**
0	[[[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘ ’]	[‘ ‘, ‘f’]]
1	[[‘ ‘, ‘ ’]	[‘ ‘, ‘ ‘]	[‘ ‘, ‘ ’]	[‘ ‘, ‘f’]]
2	[[‘ ‘, ‘ ‘]	[‘ ‘, ‘ ’]	[‘p‘, ‘f’]	[‘ ‘, ‘ ’]]
3	[[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]]]

Πίνακας 3.6 Η κατάσταση μετά την παρέμβαση του τελεστή eat

	0	**1**	**2**	**3**
0	[[[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘ ’]	[‘ ‘, ‘f’]]
1	[[‘ ‘, ‘ ’]	[‘ ‘, ‘ ‘]	[‘ ‘, ‘ ’]	[‘ ‘, ‘f’]]
2	[[‘ ‘, ‘ ‘]	[‘ ‘, ‘ ’]	[‘p‘, ‘ ’]	[‘ ‘, ‘ ’]]
3	[[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]]]

Πίνακας 3.7 Η κατάσταση μετά την παρέμβαση του τελεστή move_up

	0	**1**	**2**	**3**
0	[[[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘ ’]	[‘ ‘, ‘f’]]
1	[[‘ ‘, ‘ ’]	[‘ ‘, ‘ ‘]	[‘p‘, ‘ ’]	[‘ ‘, ‘f’]]
2	[[‘ ‘, ‘ ‘]	[‘ ‘, ‘ ’]	[‘ ‘, ‘ ’]	[‘ ‘, ‘ ’]]
3	[[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]	[‘ ‘, ‘f’]]]

Ο χώρος καταστάσεων περιλαμβάνει μία λίστα διαστάσεων 4x4 με τέσσερις λίστες κελιών. Η κάθε λίστα κελιών περιλαμβάνει τέσσερις λίστες κελιά που η κάθε λίστα κελί περιλαμβάνει δύο συμβολοσειρές, «p» ή «f» ή «” “», όπου «p» ο pacman, «f» το φρούτο και «” “» κενό. Η λίστα κελί δεν μπορεί να περιέχει δύο «p», όπως και όλη η λίστα λιστών κελιών.

4. Ο κόσμος του προβλήματος, οι τελεστές μετάβασης και η συνάρτησης εύρεσης απογόνων find_children σε γλώσσα Python

4.1 Ο κόσμος του προβλήματος

Ο κόσμος του προβλήματος σε γλώσσα Python απεικονίζεται με μία λίστα που περιλαμβάνει τέσσερις λίστες (πλέγματα κελιών), όπου η κάθε λίστα περιλαμβάνει τέσσερις λίστες-κελιά που με την σειρά τους περιλαμβάνουν δύο συμβολοσειρές, «p» ή «f» ή «” “», όπου «p» ο pacman, «f» το φρούτο και «” “» κενό. Η αρχική κατάσταση του πίνακα 1.1.1 αποθηκεύεται στην μεταβλητή init_state. Η τελική κατάσταση του πίνακα 2.2 αποθηκεύεται στην μεταβλητή goal_state. Οι καταστάσεις ορίζονται σε μία κύρια συνάρτηση «main» η οποία καλείται πρώτη από το πρόγραμμα.

Αρχική και τελική κατάσταση

```
def main():
    # Αρχική και τελική κατάσταση του προβλήματος σε διαστάσεις 4x4 (προτιμάται ο DFS
για κύριο αλγόριθμο αναζήτησης)

    init_state = [[[' ', 'f'], [' ', 'f'], [' ', ' '], [' ', 'f']],
                  [[' ', ' '], [' ', ' '], ['p', ' '], [' ', 'f']],
                  [[' ', ' '], [' ', 'f'], [' ', 'f'], [' ', ' ']],
                  [[' ', 'f'], [' ', 'f'], [' ', 'f'], [' ', 'f']]]

    goal_state = [[[' ', ' '], [' ', ' '], [' ', ' '], [' ', ' ']],
                  [[' ', ' '], [' ', ' '], [' ', ' '], [' ', ' ']],
```

```
                [[' ', ' '], [' ', ' '], [' ', ' '], [' ', ' ']],
                [[' ', ' '], [' ', ' '], [' ', ' '], ['p', ' ']]]
```

Στη συνέχεια, επιλέγεται ένας αλγόριθμος αναζήτησης από τον χρήστη, αυτός της Διάσχισης κατά Βάθος (DFS) ή αυτός της Διάσχισης κατά Πλάτος (BFS).

Επιλογή αλγορίθμου αναζήτησης

```
   method = 'Blind searching algorithm'
   # Επιλογή ενός από τους δύο αλγόριθμους αναζήτησης DFS ή BFS για τον κύριο
αλγόριθμο αναζήτησης επίλυσης του προβλήματος με τον pacman
   while method != 'DFS' and method != 'BFS':
      method = input('Choose between DFS and BFS as main blind searching algorithm :
')
```

Πραγματοποιείται η κλήση της find_solution που αποθηκεύει την αρχική κατάσταση σε μία λίστα για το μέτωπο αναζήτησης (κλήση της make_front) και σε μία λίστα λιστών για την ουρά αναζήτησης του προεπιλεγμένου αλγορίθμου αναζήτησης (κλήση της make_queue). Επιπρόσθετα, παίρνει σαν ορίσματα ένα κενό σύνολο που είναι το κλειστό σύνολο (closed) με τους κόμβους-καταστάσεις που έχουν επισκεφθεί, την τελική κατάσταση (goal_state) και τον προεπιλεγμένο αλγόριθμο αναζήτησης (method).

Κλήση της find_solution

```
# Κλήση της find_solution για την εύρεση μονοπατιού που θα οδηγήσει από την
προεπιλεγμένη αρχική στην τελική κατάσταση με βάση προεπιλεγμένο αλγόριθμο
αναζήτησης
   print (' ')
   print ('Begin Searching...')
   print ('Searching algorithm :',method)
   print (' ')
   find_solution(make_front(init_state), make_queue(init_state), [], goal_state,
method)
```

Κώδικας των συναρτήσεων make_front και make_queue

```
""" Αρχικοποίηση του μετώπου αναζήτησης """
def make_front(state):
   return [state]

""" Αρχικοποίηση της ουράς αναζήτησης """
def make_queue(state):
    return [[state]]
```

Η find_solution καλείται αναδρομικά μέχρι να βρεθεί μονοπάτι μέσω των τελεστών μετάβασης που οδηγεί από την αρχική στην τελική κατάσταση. Εάν ο τελευταίος κόμβος ενός μονοπατιού είναι η τελική κατάσταση, δηλαδή, αν ο πρώτος κόμβος-κατάσταση του μετώπου αναζήτησης είναι η τελική κατάσταση, τότε βρέθηκε το τελικό μονοπάτι και ο στόχος (κλήση της is_goal_state).

Κώδικας της συνάρτησης find_solution

```
""" Συνάρτηση εύρεσης μονοπατιού που οδηγεί από την αρχική κατάσταση στην τελική
μέσω των αλγορίθμων αναζήτησης DFS ή BFS """
def find_solution(front, queue, closed, goal, method):
   if not front:
      print('No solution found! End of searching...')
   elif front[0] in closed:
        new_front = copy.deepcopy(front)
        new_front.pop(0)
        new_queue = copy.deepcopy(queue)
        new_queue.pop(0)
        find_solution(new_front, new_queue, closed, goal, method)
   elif is_goal_state(front[0]):
      print('Goal found!')
      for i in range(len(queue[0])):
            for j in range(len(queue[0][i])):
               print(queue[0][i][j])              # Οι κόμβοι-καταστάσεις στο
τελικό μονοπάτι απεικονίζονται δισδιάστατα και ο ένας κάτω από τον άλλον
            print (' ')
   else:
      closed.append(front[0])
      front_copy = copy.deepcopy(front)
      front_children = expand_front(front_copy, method)
      queue_copy = copy.deepcopy(queue)
      queue_children = extend_queue(queue_copy, method)
      closed_copy = copy.deepcopy(closed)
      find_solution(front_children, queue_children, closed_copy, goal, method)
```

Κώδικας της συνάρτησης is_goal_state

```
""" Έλεγχος εύρεσης τελικής κατάστασης """
def is_goal_state(state):
   for row in range(len(state)):
      for col in range(len(state[row])):
         if state[row][col][1] == 'f':
            return 0
   return 1
```

Το μονοπάτι βρίσκεται με την κλήση της expand_front και της extend_queue, όπου ανάλογα τον προεπιλεγμένο αλγόριθμο αναζήτησης επεκτείνεται το μέτωπο αναζήτησης και η ουρά αναζήτησης. Πιο συγκεκριμένα, στον Διάσχιση κατά Βάθος (DFS) ο κόμβος-κατάσταση που επισκέπτεται, αφαιρείται από το μέτωπο αναζήτησης και προστίθεται στο κλειστό σύνολο (closed) και στην θέση του μπαίνουν οι καταστάσεις-παιδιά του στην αρχή του μετώπου (LIFO), ενώ στον Διάσχιση κατά Πλάτος τα παιδιά μπαίνουν στο τέλος του μετώπου (FIFO).

Παρόμοια, στον Διάσχιση κατά Βάθος (DFS) το μονοπάτι που επισκέπτεται, αφαιρείται από την ουρά αναζήτησης και ο τερματικός κόμβος-κατάσταση προστίθεται στο κλειστό σύνολο (closed) και στην θέση της ουράς μπαίνουν τα μονοπάτια με τερματικούς κόμβους τις καταστάσεις-παιδιά του κόμβου που αφαιρέθηκε στην αρχή της ουράς (LIFO), ενώ στον Διάσχιση κατά Πλάτος τα παιδιά μπαίνουν στο τέλος της ουράς (FIFO).

Κώδικας της συνάρτησης expand_front

```
""" Επέκταση του μετώπου αναζήτησης """
def expand_front(front, method):
   if method == 'DFS':                          # Επέκταση του μετώπου αναζήτησης του αλγορίθμου αναζήτησης DFS
      if front:
         print("Front : ")
         for i in range(len(front)):
            for j in range(len(front[i])):
               print(front[i][j])               # Οι κόμβοι-καταστάσεις στο μέτωπο αναζήτησης απεικονίζονται δισδιάστατα και ο ένας κάτω από τον άλλον
            print (' ')
         print('_____________________________________')
         node = front.pop(0)                    # Αφαίρεση του πρώτου κόμβου-κατάσταση από το μέτωπο αναζήτησης
         for child in find_children(node):      # Εύρεση καταστάσεων-παιδιά του κόμβου που αφαιρέθηκε από το μέτωπο, μέσω της συνάρτησης find_children
            front.insert(0, child)              # Οι καταστάσεις-παιδιά του κόμβου που αφαιρέθηκε, τοποθετούνται σε δομή στοίβας (LIFO) στην αρχή του μετ΄΄ωπου αναζήτησης
   elif method == 'BFS':                        # Επέκταση του μετώπου αναζήτησης του αλγορίθμου αναζήτησης BFS
      if front:
         print ("Front : ")
         for i in range(len(front)):
            for j in range(len(front[i])):
               print(front[i][j])               # Οι κόμβοι-καταστάσεις στο μέτωπο αναζήτησης απεικονίζονται δισδιάστατα και ο ένας κάτω από τον άλλον
            print(' ')
         print('_____________________________________')
         node = front.pop(0)                    # Αφαίρεση του πρώτου κόμβου-κατάσταση από το μέτωπο αναζήτησης
         for child in find_children(node):      # Εύρεση καταστάσεων-παιδιά του κόμβου που αφαιρέθηκε από το μέτωπο, μέσω της συνάρτησης find_children
            front.append(child)                 # Οι καταστάσεις-παιδιά του κόμβου που αφαιρέθηκε, τοποθετούνται σε δομή ουράς (FIFO) στο τέλος του μετώπου αναζήτησης

   return front
```

Κώδικας της συνάρτησης extend_queue

```
""" Επέκταση της ουράς αναζήτησης """
def extend_queue(queue, method):
# Επέκταση της ουράς αναζήτησης του αλγορίθμου αναζήτησης DFS
   if method == 'DFS':
      print ("Queue : ")
      for i in range(len(queue)):
         for j in range(len(queue[i])):
```

```
            for w in range(len(queue[i][j])):
                print(queue[i][j][w])           # Οι κόμβοι-καταστάσεις στην ουρά
αναζήτησης απεικονίζονται δισδιάστατα και ο ένας κάτω από τον άλλον
            print (' ')                         # Ένα μονοπάτι είναι μία σειρά από
κόμβους ο ένας κάτω από τον άλλον και τελειώνει στον κόμβο που εμφανίζεται πριν το
μήνυμα "End of path"
         print ('------------ End of path ------------')
         print (' ')
      print('_________________________________________')
      node = queue.pop(0)                       # Αφαίρεση του πρώτου μονοπατιού από
την ουρά αναζήτησης
      queue_copy = copy.deepcopy(queue)
      children = find_children(node[-1])        # Εύρεση καταστάσεων-παιδιά του
κόμβου που βρισκόταν στο τέλος του μονοπατιού που αφαιρέθηκε από την ουρά, μέσω της
κλήσης find_children
      for child in children:
         path = copy.deepcopy(node)
         path.append(child)                     # Οι καταστάσεις-παιδιά αποθηκεύονται
στη λίστα path δημιουργώντας ένα νέο μονοπάτι με τερματισμό έναν κόμβο-παιδί του
τελευταίου κόμβου που βρισκόταν στο μονοπάτι που αφαιρέθηκε από την ουρά
         queue_copy.insert(0,path)              # Το μονοπάτι τοποθετείται σε δομή
στοίβας (LIFO), στην αρχή της ουράς αναζήτησης
# Επέκταση της ουράς αναζήτησης του αλγορίθμου αναζήτησης BFS
   elif method == 'BFS':
      print ("Queue : ")
      for i in range(len(queue)):
         for j in range(len(queue[i])):
            for w in range(len(queue[i][j])):
               print(queue[i][j][w])            # Οι κόμβοι-καταστάσεις στην ουρά
αναζήτησης απεικονίζονται δισδιάστατα και ο ένας κάτω από τον άλλον
            print (' ')
         print ('------------ End of path ------------')
         print (' ')
      print('_________________________________________')
      node = queue.pop(0)                       # Αφαίρεση του πρώτου μονοπατιού από
την ουρά αναζήτησης
      queue_copy = copy.deepcopy(queue)
      children = find_children(node[-1])        # Εύρεση καταστάσεων-παιδιά του
κόμβου που βρισκόταν στο τέλος του μονοπατιού που αφαιρέθηκε από την ουρά, μέσω της
κλήσης find_children
      for child in children:
         path = copy.deepcopy(node)
         path.append(child)                     # Οι καταστάσεις-παιδιά αποθηκεύονται
στη λίστα path δημιουργώντας ένα νέο μονοπάτι με τερματισμό έναν κόμβο-παιδί του
τελευταίου κόμβου που βρισκόταν στο μονοπάτι που αφαιρέθηκε από την ουρά
         queue_copy.append(path)                # Το μονοπάτι τοποθετείται σε δομή
ουράς (FIFO), στο τέλος της ουράς αναζήτησης
```

```
    return queue_copy
```

4.2 Οι τελεστές μετάβασης

Ο κάθε τελεστής μετάβασης προκειμένου να γίνει η χρήση του από τον pacman θα πρέπει να τηρεί κάποιες προϋποθέσεις. Τις προϋποθέσεις αυτές τις ελέγχουν οι συναρτήσεις can_x (όπου x ο τελεστή μετάβασης). Η λειτουργία και οι προϋποθέσεις χρήσης του κάθε τελεστή αναλύονται στον πίνακα 2.1 Οι συναρτήσεις των τελεστών επιστρέφουν τις καστάσεις έπειτα από παρέμβαση τους ή την ίδια κατάσταση που δεχτήκανε σαν όρισμα, καθώς δεν τηρείται η προβλεπόμενη προϋπόθεση.

Κώδικας της συνάρτησης can_eat

```
""" Έλεγχος χρήσης του τελεστή φαγώματος φρούτου """
def can_eat(state):
    for row in range(len(state)):
        for col in range(len(state[row])):
            if state[row][col][0] == 'p' and state[row][col][1] == 'f':
                return 1
    return 0
```

Κώδικας της συνάρτησης can_move_right

```
""" Έλεγχος χρήσης του τελεστή μετακίνησης προς τα δεξιά """
def can_move_right(state):
    for row in range(len(state)):
        if state[row][len(state[row]) - 1][0] == 'p':
            return 0
    return 1
```

Κώδικας της συνάρτησης can_move_left

```
""" Έλεγχος χρήσης του τελεστή μετακίνησης προς τα αριστερά """
def can_move_left(state):
    for row in range(len(state)):
        if state[row][0][0] == 'p':
            return 0
    return 1
```

Κώδικας της συνάρτησης can_move_up

```
""" Έλεγχος χρήσης του τελεστή μετακίνησης προς τα πάνω """
def can_move_up(state):
    for col in range(len(state[0])):
        if state[0][col][0] == 'p':
            return 0
    return 1
```

Κώδικας της συνάρτησης can_move_down

```
""" Έλεγχος χρήσης του τελεστή μετακίνησης προς τα κάτω """
def can_move_down(state):
   for col in range(len(state[len(state) - 1])):
      if state[len(state) - 1][col][0] == 'p':
         return 0
   return 1
```

Κώδικας της συνάρτησης eat

```
""" Τελεστής φαγώματος φρούτου """
def eat(state):
   if can_eat(state):
      for row in range(len(state)):
         for col in range(len(state[row])):
            if state[row][col][0] == 'p' and state[row][col][1] == 'f':
               state[row][col][1] = ' '
               return state
   else:
      return state
```

Κώδικας της συνάρτησης move_right

```
""" Τελεστής μετακίνησης προς τα δεξιά """
def move_right(state):
   if can_move_right(state):
      for row in range(len(state)):
         for col in range(len(state[row])):
            if state[row][col][0] == 'p':
               state[row][col][0] = ' '
               state[row][col + 1][0] = 'p'
               return state
   else:
      return state
```

Κώδικας της συνάρτησης move_left

```
""" Τελεστής μετακίνησης προς τα αριστερά """
def move_left(state):
   if can_move_left(state):
      for row in range(len(state)):
         for col in range(len(state[row])):
            if state[row][col][0] == 'p':
               state[row][col][0] = ' '
               state[row][col - 1][0] = 'p'
               return state
   else:
      return state
```

Κώδικας της συνάρτησης move_up

```
""" Τελεστής μετακίνησης προς τα πάνω """
def move_up(state):
   if can_move_up(state):
      for row in range(len(state)):
         for col in range(len(state[row])):
            if state[row][col][0] == 'p':
               state[row][col][0] = ' '
               state[row - 1][col][0] = 'p'
               return state
   else:
      return state
```

Κώδικας της συνάρτησης move_down

```
""" Τελεστής μετακίνησης προς τα κάτω """
def move_down(state):
   if can_move_down(state):
      for row in range(len(state)):
         for col in range(len(state[row])):
            if state[row][col][0] == 'p':
               state[row][col][0] = ' '
               state[row + 1][col][0] = 'p'
               return state
   else:
      return state
```

4.3 Η συνάρτησης εύρεσης απογόνων find_children

Η συνάρτηση εύρεσης απογόνων find_children αποθηκεύει σε μία λίστα children τις καταστάσεις-παιδιά που δημιουργούν οι πέντε προαναφερόμενοι τελεστές εφόσον τηρούνται οι προϋποθέσεις. Πραγματοποιούνται οι κλήσεις των τελεστών μετάβασης που επιστρέφουν τις καταστάσεις που δημιούργησαν με την παρέμβαση τους ή την ίδια κατάσταση λόγω μη τήρησης των προϋποθέσεων.

Κώδικας της συνάρτησης find_children

```
""" Συνάρτηση εύρεσης απογόνων μίας κατάστασης-κόμβος """
def find_children(state):
   children = []
   right_state = copy.deepcopy (state)
   child_right = move_right (right_state)      # Εύρεση απογόνου-καταστάση με την
οποία ο pacman μπορεί να μετακινηθεί προς τα δεξιά
   left_state = copy.deepcopy (state)
   child_left = move_left (left_state)         # Εύρεση απογόνου-καταστάση με την
οποία ο pacman μπορεί να μετακινηθεί προς τα αριστερά
   up_state = copy.deepcopy (state)
   child_up = move_up (up_state)               # Εύρεση απογόνου-καταστάση με την
οποία ο pacman μπορεί να μετακινηθεί προς τα πάνω
```

```
    down_state = copy.deepcopy (state)
    child_down = move_down (down_state)          # Εύρεση απογόνου-καταστάση με την
οποία ο pacman μπορεί να μετακινηθεί προς τα κάτω
    eat_state = copy.deepcopy (state)
    child_eat = eat (eat_state)                  # Εύρεση απογόνου-καταστάση με την
οποία ο pacman μπορεί να φάει φρούτο

    if not child_right == None:                  # Έλεγχος τυχόν εύρεσης απογόνου-
κατάσταση με την οποία ο pacman μπορεί να μετακινηθεί προς τα δεξιά
        children.append(child_right)
    if not child_left == None:                   # Έλεγχος τυχόν εύρεσης απογόνου-
κατάσταση με την οποία ο pacman μπορεί να μετακινηθεί προς τα αριστερά
        children.append(child_left)
    if not child_up == None:                     # Έλεγχος τυχόν εύρεσης απογόνου-
κατάσταση με την οποία ο pacman μπορεί να μετακινηθεί προς τα πάνω
        children.append(child_up)
    if not child_down == None:                   # Έλεγχος τυχόν εύρεσης απογόνου-
κατάσταση με την οποία ο pacman μπορεί να μετακινηθεί προς τα κάτω
        children.append(child_down)
    if not child_eat == None:                    # Έλεγχος τυχόν εύρεσης απογόνου-
κατάσταση με την οποία ο pacman μπορεί να φάει φρούτο
        children.append(child_eat)
```

5. Αποτελέσματα του αλγορίθμου αναζήτησης Διάσχιση κατά Πλάτος BFS

5.1 1ο Παράδειγμα

Αρχική κατάσταση του προβλήματος μπορεί να οριστεί οποιαδήποτε κατάσταση με την οποία ο pacman και τα φρούτα βρίσκονται σε ένα κελί εντός του πλέγματος που περιλαμβάνει πλέγματα με κελιά. Για παράδειγμα σε πλέγμα με διαστάσεις 2x2 αρχική κατάσταση μπορεί να οριστεί ως εξής :

Πίνακας 5.1.1 Η αρχική κατάσταση

	0	**1**
0	[[[‘ ‘, ‘f’]	[‘p‘, ‘ ’]
1	[[‘ ‘, ‘f’]	[‘ ‘, ‘ ‘]

Τελική κατάσταση του προβλήματος μπορεί να είναι οποιαδήποτε κατάσταση με την οποία ο pacman έχει φάει όλα τα φρούτα, δηλαδή, βρίσκεται μόνος του σε ένα κελί του πλέγματος. Για παράδειγμα σε πλέγμα με διαστάσεις 2x2 τελική κατάσταση μπορεί να οριστεί ως εξής:

Πίνακας 5.1.2 Η τελική κατάσταση

	0	**1**
0	[[[‘ ‘, ‘ ’]	[‘ ‘, ‘ ’]
1	[[‘p‘, ‘ ’]	[‘ ‘, ‘ ‘]

Αποτελέσματα αλγορίθμου αναζήτησης BFS για το 1ο παράδειγμα

Choose between DFS and BFS as main blind searching algorithm :

Begin Searching...

Searching algorithm : BFS

Front :

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

_____

Queue :

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

_____

Front :

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

_____

Queue :

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

---------- End of path ----------

_____

Front :

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

_____

Queue :

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

_____

Front :

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

_____

Queue :

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

_____

Front :

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

_____

Queue :

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

---------- End of path ----------

_____

Front :

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

_____

Queue :

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

---------- End of path ----------

_____

Front :

[[' ', ' '], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

Queue :

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

_____

Front :

[[' ', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

_____

Queue :

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]
[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]
[['p', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]
[[' ', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]
[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]
[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]
[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]
[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]
[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]
[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

---------- End of path ----------

_____

Front :

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', ' ']]

Queue :

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]
[['p', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]
[[' ', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]
[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]
[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]
[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]
[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]
[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]
[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]
[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', ' ']]

---------- End of path ----------

_____

Front :

[['p', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', ' '], [' ', ' ']]

Queue :

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

_____

Front :

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

Queue :

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]
[['p', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]
[[' ', 'f'], ['p', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]
[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]
[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]
[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]
[['p', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]
[[' ', 'f'], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]
[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]
[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]
[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]
[['p', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]
[['p', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]
[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]
[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]
[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]
[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]
[[' ', ' '], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]
[['p', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], ['p', ' ']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', 'f'], ['p', ' ']]

[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]
[['p', 'f'], [' ', ' ']]

[[' ', 'f'], [' ', ' ']]
[['p', ' '], [' ', ' ']]

[['p', 'f'], [' ', ' ']]
[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', ' ']]
[[' ', ' '], [' ', ' ']]

---------- End of path ----------

_____

Goal found!
[[' ', 'f'], ['p', ' ']]
[[' ', 'f'], [' ', ' ']]

[['p', 'f'], [' ', ' ']]
[[' ', 'f'], [' ', ' ']]

[['p', ' '], [' ', ' ']]
[[' ', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]
[['p', 'f'], [' ', ' ']]

[[' ', ' '], [' ', ' ']]
[['p', ' '], [' ', ' ']]

5.2 2ο Παράδειγμα

Αρχική κατάσταση του προβλήματος μπορεί να οριστεί οποιαδήποτε κατάσταση με την οποία ο pacman και τα φρούτα βρίσκονται σε ένα κελί εντός του πλέγματος που περιλαμβάνει πλέγματα με κελιά. Για παράδειγμα σε πλέγμα με διαστάσεις 2x2 αρχική κατάσταση μπορεί να οριστεί ως εξής :

Πίνακας 5.2.1 Η αρχική κατάσταση

	0	**1**
0	[[‘ ‘, ‘ ’]	[‘ ‘, ‘f’]
1	[[‘p‘, ‘ ’]	[‘ ‘, ‘f‘]

Τελική κατάσταση του προβλήματος μπορεί να είναι οποιαδήποτε κατάσταση με την οποία ο pacman έχει φάει όλα τα φρούτα, δηλαδή, βρίσκεται μόνος του σε ένα κελί του πλέγματος. Για παράδειγμα σε πλέγμα με διαστάσεις 2x2 τελική κατάσταση μπορεί να οριστεί ως εξής:

Πίνακας 5.2.2 Η τελική κατάσταση

	0	**1**
0	[[‘ ‘, ‘ ’]	[‘p‘, ‘ ’]
1	[[‘ ‘, ‘ ’]	[‘ ‘, ‘ ‘]

Αποτελέσματα αλγορίθμου αναζήτησης BFS για το 2ο παράδειγμα

Choose between DFS and BFS as main blind searching algorithm :

Begin Searching...

Searching algorithm : BFS

Front :

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

_____

Queue :

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

---------- End of path ----------

Front :

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

Queue :

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

---------- End of path ----------

Front :

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

_____

Queue :

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

---------- End of path ----------

_____

Front :

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

Queue :

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

---------- End of path ----------

_____

Front :

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

Queue :

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]
[['p', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]
[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]
[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]
[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]
[[' ', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]
[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]
[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]
[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]
[[' ', ' '], ['p', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

---------- End of path ----------

_____

Front :

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

_____

Queue :

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

_____

Front :

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', 'f']]

_____

Queue :

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', 'f']]

---------- End of path ----------

_____

Front :

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

Queue :

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

_____

Front :

[['p', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', ' ']]

Queue :

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

_____

Front :

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', 'f']]

Queue :

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', 'f']]

---------- End of path ----------

_____

Front :

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

Queue :

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', ' ']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

[['p', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

---------- End of path ----------

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

---------- End of path ----------

_____

Goal found!

[[' ', ' '], [' ', 'f']]

[['p', ' '], [' ', 'f']]

ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', 'f']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], ['p', 'f']]

[[' ', ' '], [' ', ' ']]

[[' ', ' '], ['p', ' ']]

[[' ', ' '], [' ', ' ']]

ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ

Σας ευχαριστώ για την προσοχή σας.