ΤΕΙ ΣΕΡΡΩΝ Τμήμα Πληροφορικής και Επικοινωνιών

Εργαστήριο Λειτουργικών συστημάτων ΙΙ

Τμήμα Πέμπτης 18-20

Ονοματεπώνυμο: *Ψαρίδης Αθανάσιος*

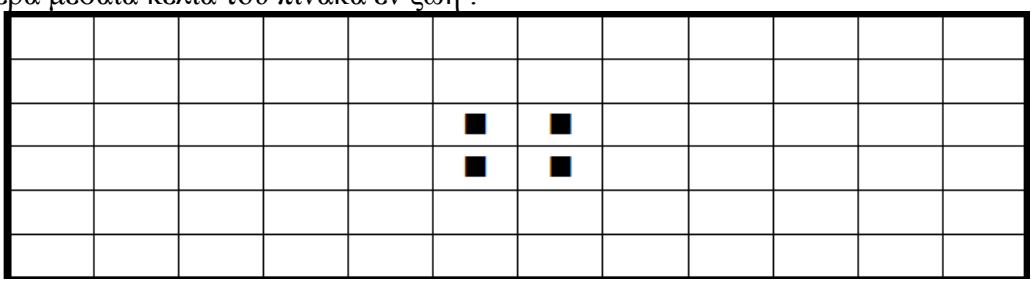
E-mail: *Psaridis@gmail.com*

ΑΜ: *1820*

Τίτλος Project: *Το παιχνίδι της ζωής (The Game of Life)*

Εισαγωγή

12. Το Παιχνίδι της Ζωής (The Game of Life). Το Project αυτό συνίσταται στην κατασκευή ενός προγράμματος σεναρίου κελύφους, που θα προσομοιώνει την εξέλιξη μίας αποικίας μονοκύτταρων οργανισμών (cells). Η προσομοίωση αυτή είναι γνωστή και ως "Το παιγνίδι της ζωής" ή "Life".
Η αποικία των cells είναι στην ουσία ένας διδιάστατος πίνακας (π.χ. 20 X 20 θέσεων) που αντιστοιχούν σε θέσεις κελιών μέσα στην αποικία. Το κάθε κελί μπορεί να είναι "εν ζωή" (1) ή όχι (0). Η εξέλιξη της ζωής στην αποικία ξεκινά με ένα συγκεκριμένο "πατρόν" από ζωντανά κελιά. Για παράδειγμα μπορεί κανείς να θέσει τα τέσσερα μεσαία κελιά του πίνακα εν ζωή :



Σε κάθε γενιά (επανάληψη) της εξέλιξης το κάθε κελί του πίνακα αποκτά ζωή ή την χάνει σύμφωνα με τους παρακάτω κανόνες :

1. Αν ένα κενό κελί γειτονεύει με τρία "ζωντανά" κελιά τότε στο κελί αυτό δημιουργείται ζωή.
2. Αν ένα κελί είναι "εν ζωή" και γειτονεύει με 2 ή 3 "ζωντανά" κελιά τότε το κελί διατηρείται
3. Αν ένα κελί είναι "εν ζωή" και γειτονεύει με 1 ή κανένα "ζωντανό" κελί τότε το κελί πεθαίνει λόγω μοναξιάς
4. Αν ένα κελί είναι "εν ζωή" και γειτονεύει με 4 ή περισσότερα "ζωντανά" κελιά τότε το κελί πεθαίνει λόγω συνωστισμού.

Η εξέλιξη συνεχίζεται μέχρι έναν συγκεκριμένο αριθμό επαναλήψεων που έχει δωθεί από τον χρήστη, ή μεχρι ο χρήστης να πατήσει CTRL+C. Στην προσομοίωση αυτή παρατηρούνται περίεργες συμπεριφορές του πληθυσμού των οργανισμών που εξαρτώνται άμεσα από το αρχικό pattern των ζωντανών κελιών. Ο αριθμός των κελιών οριζόντια και κάθετα όπως και το αρχικό pattern ζωντανών κελιών πρέπει να καθορίζονται κάθε φορά από τον χρήστη.

Τρόπος λειτουργίας

Το συγκεκριμένο πρόγραμμα υλοποιήθηκε σε περιβάλλον **windows** με την χρήση του **Cygwin** και την βοήθεια του **Notepad++** ως editor.
Η λειτουργία του script είναι πολύ απλή και μπορεί να τρέξει σχεδόν σε όλα τα **unix** συστήματα που υποστηρίζουν το φίλτρο **AWK.**
Το κυρίως πρόγραμμα βρίσκεται στο αρχείο **life.sh.** Εάν το αρχείο αυτό δεν εχει δικαιώματα εκτέλεσης στον unix σύστημα που δουλεύετε τότε θα πρέπει να του τα δώσετε εσείς χρησημοποιώντας την εντολή

chmod +x life.sh (οπου life.sh πρέπει να δίνεται το path του αρχείου life.sh εκτός αν βρίσκεται στο current directory **).**
Αφού λοιπόν δώσουμε δικαιώματα εκτέλεσης στο αρχείο είμαστε έτοιμοι να τρέξουμε την εφαρμογή.

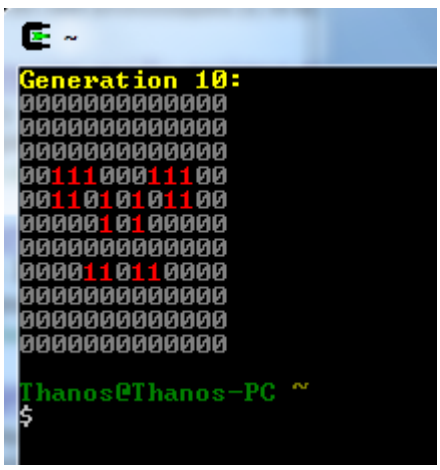
./life.sh < αρχείο

Ο τελεστής < χρησημοποιείται για Ανακατεύθυνση της εισόδου από αρχείο. Με λίγα λόγια το κυρίως πρόγραμμα περιμένει να διαβάσει δεδομένα από ένα αρχείο που θα του στείλουμε με την βοήθεια του τελεστή <
Οπου **"αρχείο"** είναι ένα αρχείο με το αρχικό pattern των ζωντανών και πεθαμένων κελιών που έχουμε ορίσει εμείς. Στο αρχείο αυτό πρέπει να ορίσουμε τα ζωντανά κελιά με τον αριθμό 1 και τα πεθαμένα κελιά με τον αριθμό 0. Έτσι δημιουργείται μια πινακοειδή μορφή η οποία θα διαβαστεί απο το πρόγραμμα και με βάση αυτή θα ξεκινήσει η εξέλιξη. Δεν πρέπει να ξεχάσουμε να σημπληρώσουμε κάποιο κελί του πίνακα είτε με 0 είτε με 1 αλλιώς το πρόγραμμα θα μας ειδοποιήσει ότι υπάρχει λάθος στην εισαγωγή των δεδομένων και θα τερματίσει.
Μαζί με την εφαρμογή life.sh παρέχονται κάποια έτοιμα αρχεία-patterns τα οποία παρουσιάζουν ιδιαίτερο ενδιαφέρον και συμμετρία καθώς εξελίσσονται.

παράδειγμα αρχείου tumbler:

```
0000000000000
0000000000000
0000110110000
0000110110000
0000010100000
0001010101000
0001010101000
0001100011000
0000000000000
0000000000000
0000000000000
```

παράδειγμα εκτέλεσης του tumbler



Ένας άλλος τρόπος για να τρέξουμε το πρόγραμμα life.sh είναι να τυπώσουμε το αρχείο δεδομένων στην οθόνη (με την βοήθεια της cat) και να διοχετεύσουμε την έξοδό του σαν είσοδο για το κυρίως πρόγραμμα.

cat αρχείο | ./life.sh

Επιπλέον επιλογές

Το πρόγραμμα .life.sh προσφέρει κάποιες επιπλέον παραμέτρους για τον αριθμό εξελίξεων και την ταχύτητα.
Θα παρατηρήσατε ότι όταν τρέξατε το πρόγραμμα έγιναν 10 εξελίξεις (generations) με ταχύτητα γύρω στα 0.5sec και μετα σταμάτησε. Αυτό συμβαίνει κάθε φορά απο προεπιλογή εκτός εαν καθορίσει ο χρήστης τον αριθμό εξελίξεων και την ταχύτητα για παράδειγμα:

./life.sh 100 0 < tumbler

Η παραπάνω εντολή θα διαβάσει το αρχείο tumbler και θα εκτελέσει 100 επαναλήψεις (generations) με ταχύτητα 0 sec. Η εκτέλεση θα γίνει αστραπιαία.

Μαζί με το κυρίως πρόγραμμα και τα αρχεία δεδομένων παρέχεται επίσης και ένα δεύτερο πρόγραμμα με όνομα **generator.sh** Αυτό το πρόγραμμα μπορεί να χρησημοποιηθεί ως γεννήτρια τυχαίων πινάκων με ζωντανά και πεθαμένα κελιά την οποία θα μπορούμε να διοχετεύσουμε στο κυρίως πρόγραμμα για εισαγωγή δεδομένων. Εάν πληκτρολογίσετε

./generator.sh
στην οθόνη θα εμφανιστεί ενας πίνακας 20x20 με τυχαίους αριθμούς 0 και 1. για να αλλάξετε το μέγεθος του πίνακα γράψτε.
./generator.sh 15 15
Αυτό δημιουργεί έναν πίνακα 15x15 τυχαίων αριθμών. για να διοχετεύσουμε τον παραπάνω πίνακα σαν είσοδο για το κύριως πρόγραμμα γράφουμε.
./generator.sh 15 15 | ./life.sh 100 0
Η παραπάνω εντολή δημιουργεί έναν πίνακα 15x15 τον διοχετεύει στο life.sh το οποίο θα εκτελέσει 100 εξελίξεις με ταχύτητα 0 sec

Επεξήγηση κώδικα

Για την δημιουργία αυτού του project χρησιμοποιήθηκε το κέλυφος bash και η AWK σαν βασικό φίλτρο. Γενικά ο κώδικας χωρίζεται σε 3 μέρη:

- Το πρώτο μέρος στο οποίο αρχικοποιούνται κάποιες μεταβλητές όπως τα ζωντανά και πεθαμένα κελια (ALIVE=1, DEAD=0) και ο αριθμός των επαναλήψεων του προγράμματος (runs) καθώς και η ταχύτητα (speed).
- Το δεύτερο μέρος στο οποίο γίνεται η ανάγνωση του αρχείου δεδομένων και τα δεδομένα απο το αρχείο αποθηκεύονται στην μεταβλητή data.
- Το τρίτο μέρος που είναι και το κύριο πρόγραμμα όπου εκεί καλούνται κάποιες συναρτήσεις για τον υπολογισμό των εξελίξεων και την εμφάνιση των αποτελεσμάτων

Πιο αναλυτικά το πρόγραμμα ξεκινάει απο την γραμμή 7 χρησημοποιόντας ένα option της AWK το -v το οποίο μας επιτρέπει να περνάμε μεταβλητές απο το κέλυφος στην AWK. Με αυτόν τον τρόπο μπορούμε να περάσουμε σαν παράμετρους τον αριθμό των επαναλήψεων και την ταχύτητα που θέλουμε πριν τρέξουμε το πρόγραμμα. ο συμβολισμός runs=${1:-10} σημαίνει ότι το πρώτο όρισμα που θα δώσει ο χρήστης θα αποθηκευτεί στη μεταβλητή runs ενώ αν δε δώσει κανένα όρισμα η μεταβλητή runs θα πάρει την τιμή 10 απο default. Με την ίδια λογική λειτουργεί και η speed.
Παρακάτω αρχίζει το πρότυπο BEGIN {} όπου αρχικοποιούνται οι μεταβλητές ALIVE=1 DEAD=0 και τυπώνεται στην οθόνη μια spash screen με χρώματα χρησημοποιώντας ακολουθίες διαφυγής τύπου ANSI. (γραμμές 8-19). Απο γραμμή 21-30 και 33-39 είναι κάποιες συναρτήσεις οι οποίες θα καλεστούν αργότερα.
Η συνάρτηση checkAlive() χρησιμοποιήται για τον έλεγχο των ζωντανών γειτονικών κελιών που βρίσκονται μέσα στα σύνορα του πίνακα. Η συνάρτηση countAlive() καλεί την checkAlive() και επιστρέφει το συνολικό πλήθος των γειτονικών κελιών. Η συνάρτηση display() χρησιμοποιείται για να τυπώνει τον πίνακα στην οθόνη. τα ζωντανά κελιά αναπαρίστανται με 1 και κόκκινο χρώμα ενώ τα πεθαμένα με 0 και μαύρο χρώμα.
Απο τις γραμμές 55-59 αρχίζει η ανάγνωση του αρχείο δεδομένων ο κώδικας που βρίσκεται ανάμεσα στα { } θα εκτελεστεί τόσες φορές όσες και οι γραμμές του αρχείου που διαβάζεται τα δεδομένα του αρχείου αποθηκεύονται στην data και ταυτόχρονα μετρούνται και οι γραμμές του αρχείου (γραμμές πίνακα) με την lines. Μετά εκτελείται ο κώδικας END{} μέχρι το τέλος όπου εδώ καλούνται οι παραπάνω συναρτήσεις. Και ελέγχεται το πλήθος των γειτονικών κελιών που περικλείουν ένα κελί κάθε φορά. Εάν ο αριθμός των γειτονικών ζωντανών κελιών ειναι 2 τότε δεν γίνεται καμία αλλαγή ουσιαστικά στην παλιά κατάσταση. Εάν ο αριθμός είναι 3 τότε δημιουργείται ζωη στην καινούρια κατάσταση ή διατηρείται η ζωή. Όλες οι άλλες περιπτώσεις καταλήγουν σε πεθαμένα κελιά στην καινούρια κατάσταση. Τυπώνεται το αποτέλεσμα και τα παραπάνω βήματα εκτελούνται τόσες φορές όσες και ο αριθμός επαναλήψεων που έχει δωθεί (runs). Επίσης στο πρόγραμμα χρησημοποιείται η έτοιμη συνάρτηση substr(string,start,end) η οποία χρησημοποιείται για να εξαγάγει ένα υπο-string απο ένα κανονικό string αρχίζοντας απο τον αριθμό start και τελιώνοντας στο end. Πχ
data = "hello"; data2 = substr(data,1,4); Σε αυτή την περίπτωση το data2 έχει την τιμή "hell"