ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ
ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΔΠΜΣ – ΡΟΗ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΑΝΑΠΤΥΞΗ ΚΑΙ ΣΧΕΔΙΑΣΗ ΛΟΓΙΣΜΙΚΟΥ

Η γλώσσα προγραμματισμού C

ΕΡΓΑΣΤΗΡΙΟ 1.1: Βασικά στοιχεία +

Μάρτιος 2019

Το σημερινό εργαστήριο αποτελεί συνέχεια του Εργαστηρίου 1, εμβαθύνοντας σε εντολές ελέγχου και εκφράσεις της γλώσσας C. Οι εντολές ελέγχου είναι αυτές που τροποποιούν τη σειριακή ροή του κώδικα, εκτελώντας άλματα προς άλλο σημείο του κώδικα από αυτό που ακολουθεί, είτε με κάποια συνθήκη, είτε χωρίς συνθήκη. Από τις εντολές ελέγχου ροής με συνθήκη, θα συνεχίσετε την εξάσκηση στη χρήση της εντολής *if*, και θα δείτε και την εντολή επιλογής *switch* καθώς και την εντολή επανάληψης με συνθήκη *while*.

Η παρακάτω άσκηση αποτελεί μέρος των πιο προχωρημένων εργασιών του μαθήματος και μπορεί να παραδοθεί μέχρι το τέλος του εξαμήνου.

Άσκηση: «Ένα απλό παιχνίδι»

Στην άσκηση αυτή θα γράψετε ένα πρόγραμμα που θυμίζει παιδικό παιχνίδι, όπου σε κάποιο ταμπλό προσπαθείτε από μια αφετηρία να φτάσετε σε έναν τερματισμό. Έτσι λοιπόν, στο παιχνίδι αυτό θα χρησιμοποιήσετε ένα ζάρι, ένα δείκτη θέσης, και θα προχωράτε μπρος και πίσω, ανάλογα με την τιμή που δείχνει το ζάρι. Το παιχνίδι μετράει πόσες κινήσεις χρειαστήκατε για να περάσετε τη θέση τερματισμού. Μπορείτε να συγκρίνετε τον αριθμό κινήσεων με το διπλανό σας για να βγάλετε νικητή.

Το βασικό στοιχείο που χρειάζεται το παιχνίδι είναι το τυχαίο, ώστε να μπορέσετε να πετύχετε τη λειτουργία του ζαριού. Στους υπολογιστές δεν είναι δυνατό να επιτευχθεί πραγματική τυχαιότητα στους αριθμούς, μια που η κανονική λειτουργία ενός υπολογιστή είναι απόλυτα προβλέψιμη. Όμως υπάρχει η ψευδοτυχαιότητα, η οποία – αν και προβλέψιμη – προσεγγίζει ικανοποιητικά την τυχαιότητα. Συγκεκριμένα, μπορεί να παραχθεί μια ακολουθία αριθμών μέσω κάποιας μαθηματικής συνάρτησης, στην οποία οι αριθμοί μεταβάλλονται με τρόπο που μοιάζει με τυχαίο. Λόγω της φύσης του υπολογιστή, η ακολουθία αυτή θα είναι περιοδική, άρα όχι πραγματικά τυχαία. Αν όμως η περίοδος επανάληψης είναι πολύ μεγάλη, τότε η συμπεριφορά της ακολουθίας είναι σχεδόν τυχαία, ή αλλιώς *ψευδοτυχαία*.

Η βιβλιοθήκη *stdlib* παρέχει κάποιες συναρτήσεις τυχαίων αριθμών τις οποίες και θα μάθετε να χρησιμοποιείτε. Έτσι, η συνάρτηση *rand()* – χωρίς παραμέτρους – επιστρέφει έναν ψευδοτυχαίο ακέραιο αριθμό από 0 έως *RAND_MAX*, όπου *RAND_MAX* σταθερά δηλωμένη στη βιβλιοθήκη. Δυστυχώς, σε κάθε εκκίνηση ενός προγράμματος, η συνάρτηση αυτή ξεκινάει από το ίδιο σημείο της ακολουθίας, οπότε αν και σε ψευδοτυχαία σειρά, η συμπεριφορά του κώδικα θα είναι κάθε φορά η ίδια. Η συνάρτηση *srand()* με παράμετρο ένα μη προσημασμένο ακέραιο αριθμό καθορίζει

το σημείο εκκίνησης της ακολουθίας, ώστε να μπορείτε να ξεκινάτε από διαφορετικό σημείο κάθε φορά. Η πιο συνηθισμένη τιμή που δίνεται στην παράμετρο της συνάρτησης αυτής είναι η τιμή του ρολογιού του επεξεργαστή, η οποία θα είναι γενικά διαφορετική κάθε φορά. Για να διαβάσετε την τιμή του ρολογιού, χρησιμοποιήστε τη συνάρτηση *time()* – με παράμετρο 0 – από τη βιβλιοθήκη *time*, την οποία πρέπει να τη δηλώσετε μαζί με τις υπόλοιπες βιβλιοθήκες στην αρχή του προγράμματος.

Ξεκινήστε ένα νέο πρόγραμμα με το γνωστό πια τρόπο. Ονομάστε το αρχείο του προγράμματος αυτής της άσκησης *lab1.4.c*. Δηλώστε τις βιβλιοθήκες *stdio* και *stdlib* με το γνωστό τρόπο αν δημιουργήσατε νέο αρχείο, αλλά και τη βιβλιοθήκη *time* με την οδηγία "#include <time.h>". Δηλώστε τη συνάρτηση *main()*.

Μέσα στη συνάρτηση *main()* θα πρέπει να αρχικοποιήσετε την ακολουθία των ψευδοτυχαίων αριθμών με την εντολή:

```
srand((unsigned int) time(0));
```

η οποία καλεί την *srand()* δίνοντας ως όρισμα την τιμή του ρολογιού που λαμβάνεται με την *time()*, με μετατροπή της τιμής αυτής σε μη προσημασμένο ακέραιο που θέλει η *srand()*. Ο τρόπος με τον οποίο κάνουμε μετατροπή τύπου, δίνοντας απλά τον τύπο σε παρενθέσεις πριν από την τιμή που θέλουμε να μετατρέψουμε, είναι πολύ συνηθισμένος στη C και θα τον χρησιμοποιήσετε πολύ συχνά στο μέλλον.

Μέχρι τώρα έχετε εξασφαλίσει ότι η ακολουθία ψευδοτυχαίων αριθμών θα διαφοροποιείται σε κάθε εκτέλεση του προγράμματος. Στη συνέχεια, πρέπει να μάθετε να ρίχνετε το ζάρι! Χρησιμοποιήστε τη συνάρτηση rand() για να πάρετε έναν αριθμό από το 1 έως το 6, ως εξής[1]:

```
d = 1 + ((long long)6*rand())/(1+(long long)RAND_MAX);
```

Το πηλίκο μετατρέπει τον τυχαίο αριθμό που κυμαίνεται από 0 έως RAND_MAX σε τυχαίο αριθμό από 0 έως 5, οπότε η πρόσθεση του 1 δίνει τιμή ζαριού. Η μετατροπή σε long long γίνεται ώστε να αποφευχθεί πιθανή υπερχείλιση στην πράξη.

Δηλώστε στην αρχή της main() την ακέραια μεταβλητή *d* όπου θα αποθηκεύετε την τιμή του ζαριού, την ακέραια μεταβλητή *board* όπου θα αποθηκεύετε τη θέση στο ταμπλό και την ακέραια μεταβλητή *moves* με την οποία θα μετράτε τις κινήσεις σας. Μετά τις δηλώσεις, ή μέσα στις δηλώσεις, αρχικοποιήστε τις δύο τελευταίες μεταβλητές στο 0. Ο τερματισμός του παιχνιδιού σηματοδοτείται με τιμή της μεταβλητής board μεγαλύτερη από 100. Στο σημείο αυτό αντιγράψτε τον πιο πάνω κώδικα, ώστε να περάσει μια ζαριά, η οποία χρησιμοποιεί παλιά τιμή τυχαίου αριθμού.

Για να υλοποιήσετε τώρα το παιχνίδι, θα πρέπει να δημιουργήσετε μια δομή επανάληψης. Χρησιμοποιήστε την εντολή *while*, δημιουργώντας για αυτήν μια σύνθετη εντολή, όπου θα βάλετε τις εντολές που περιγράφονται στη συνέχεια.

Στην αρχή της σύνθετης εντολής της *while* βάλτε μια εντολή αύξησης του αριθμού κινήσεων κατά 1.

Εισάγετε ξανά τον παραπάνω κώδικα ρίψης ζαριού, και στη συνέχεια, γράψτε μια εντολή επιλογής *switch*, η οποία για τις διάφορες τιμές της μεταβλητής *d* να υλοποιεί κάποια κίνηση στο ταμπλό του παιχνιδιού. Για παράδειγμα, μπορείτε για 1 να αυξάνετε τη μεταβλητή *board* κατά 1, για 2 να διπλασιάζετε την τιμή της *board*, για 3 να

[1] Άλλη συνηθισμένη και πιο απλή τεχνική για λήψη ψευδοτυχαίων αριθμών από 0 μέχρι N είναι με το υπόλοιπο: `rand()%N`, όμως η τεχνική αυτή δεν δίνει καλή τυχαιότητα στην ακολουθία αριθμών που παράγει.

επαναφέρετε την τιμή της *board* στο 0, για 4 να ξαναρίχνετε το ζάρι, για 5 να μειώνετε την *board* κατά 2, και για 6 να προσθέτετε 10 στην τιμή της *board*. Μην ξεχνάτε να κλείνετε κάθε επιλογή με εντολή *break*, εκτός αν θέλετε σκόπιμα να συνεχίζετε με τον κώδικα της επόμενης επιλογής.

Ειδικά για να ξαναρίξετε το ζάρι, θα πρέπει να επαναλάβετε όλη τη διαδικασία, χωρίς όμως να αυξήσετε το μετρητή κινήσεων. Για να το επιτύχετε αυτό, δηλώστε στην αρχή της *main()* μία ακόμα τοπική μεταβλητή, έστω *repeat*, η οποία να λειτουργεί ως σημαία, δηλαδή να παίρνει τιμές 0 και 1. Έτσι, στην επιλογή επανάληψης ρίψης του ζαριού, θα δώσετε 1 στη μεταβλητή αυτή, ενώ η αύξηση κινήσεων θα γίνεται μέσα από μία εντολή *if* η οποία θα ελέγχει αν η μεταβλητή *repeat* έχει τιμή 0, ώστε μόνο τότε να επιτρέπει την αύξηση. Αρχικοποιήστε τη μεταβλητή αυτή σε 0, ώστε το πρόγραμμα να αρχίσει κανονικά να μετράει τις επαναλήψεις, και φροντίστε αυτή να μηδενίζεται μετά την εντολή *if*, ώστε η επόμενη ρίψη του ζαριού να μετρήσει ως νέα κίνηση, εκτός βέβαια αν ξαναέρθει η ίδια τιμή ζαριού!

Σε κάθε τιμή του ζαριού, προσθέστε εκτύπωση κατάλληλου μηνύματος που να ενημερώνει τον παίκτη για την τιμή αυτή και για την κίνηση που γίνεται με αυτή. Ένα τέτοιο μήνυμα, για τιμή 3 – σε συμφωνία με τα παραπάνω, θα μπορούσε να είναι: "You are very unlucky! You got a 3, and you are sent back to position 0...\n".

Μετά την εντολή *switch* θα πρέπει να ελέγχετε αν το ταμπλό ξεπέρασε το σημείο τερματισμού. Εισάγετε μια εντολή *if* η οποία να ελέγχει αν η μεταβλητή *board* έχει τιμή μεγαλύτερη από 100, ώστε σε περίπτωση αληθούς συνθήκης, να εκτυπώνει κατάλληλο μήνυμα τερματισμού (πχ. "Game Over\n"), όπου όμως να αναγράφεται και το σκορ, δηλαδή ο αριθμός κινήσεων που έγιναν. Εδώ θα πρέπει να τερματίζεται η δομή επανάληψης της εντολής *while*. Μπορείτε να χρησιμοποιήσετε την εντολή *break*, ή να ορίσετε κατάλληλη μεταβλητή ελέγχου του βρόχου *while*, η οποία να χρησιμοποιείται στην έκφραση συνθήκης της *while*.

Μετά την εντολή *while*, στο τέλος του προγράμματος, προσθέστε την εντολή επιστροφής «return 0;». Μεταφράστε και εκτελέστε τον κώδικά σας.

Θα παρατηρήσατε ότι το πρόγραμμα εκτελέστηκε υπερβολικά γρήγορα για να το παρακολουθήσετε. Προσθέστε λοιπόν δύο εντολές στο τέλος της *while*, μια εκτύπωσης του μηνύματος “Press Enter to continue”, και μία ανάγνωσης με κλήση της συνάρτησης *getchar()* χωρίς αποθήκευση του αποτελέσματός της.

Τώρα θα διαπιστώσετε ότι από τα πολλά Enter χάνετε την τελευταία εκτύπωση του προγράμματος! Για να λύσετε αυτό το πρόβλημα, βάλτε μια εντολή ανάγνωσης κάποιου συγκεκριμένου χαρακτήρα πριν την έξοδο, με κατάλληλη υπόδειξη. Εδώ θα πρέπει να χρησιμοποιήσετε μια δομή «while (1)», ώστε να διαβάζετε επαναληπτικά μέχρι να έρθει αυτός ο χαρακτήρας. Μπορείτε για παράδειγμα να ζητήσετε το ‘q’ (από το αγγλικό quit), και μετά το Enter. Έτσι το πρόγραμμα δεν θα κλείσει το παράθυρο χωρίς να έχετε διαβάσει το αποτέλεσμα. Ίσως να χρειαστεί να βάλετε μια πρόσθετη *getchar()* για κατανάλωση του Enter!

Ξαναμεταφράστε, επαληθεύστε, αποθηκεύστε και υποβάλετε το πρόγραμμά σας.