ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ
Τμήμα Πληροφορικής

ΕΠΛ 232 – Προγραμματιστικές Τεχνικές και Εργαλεία
ΑΣΚΗΣΗ 1 – The Kenken Puzzle

Διδάσκων: Δημήτρης Ζεϊναλιπούρ

Υπεύθυνοι Εργαστήριων: Πύρρος Μπράτσκας & Παύλος Αντωνίου

Ημερομηνία Ανάθεσης: Τέταρτη, 12 Σεπτεμβρίου 2018
Ημερομηνία Παράδοσης: Παρασκευή, 21 Σεπτεμβρίου 2018, ώρα 14:00
(ο κώδικας να υποβληθεί σε zip μέσω του Moodle)
http://www.cs.ucy.ac.cy/courses/EPL232

I. Στόχος

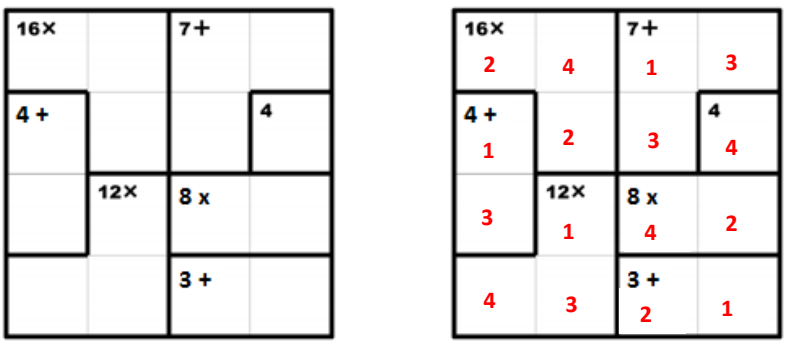
Στην εργασία αυτή θα ασχοληθούμε με βρόγχους, ελέγχους, στατικούς δισδιάστατους πίνακες, συναρτήσεις, μορφοποιημένη είσοδο/έξοδο στην οθόνη, είσοδο/έξοδο σε αρχεία. Επίσης θα γίνει η επί μέρους χρήση των πιο κάτω εργαλείων: ο οδηγός σχολίων θα πρέπει να δοθεί σε **doxygen**, χρήση της **ulimit** για ενδεχομένη προσαρμογή της στοίβας του προγράμματος και η **time** για εύρεση του χρόνου εκτέλεσης του προγράμματος.

II. Περιγραφή του Παιχνιδιού

Για αυτήν την άσκηση πρέπει να υλοποιήσετε ένα παιχνίδι, με όνομα Kenken puzzle, το οποίο είναι μια παραλλαγή του γνωστού παιχνιδιού Sudoku. Όπως και στο Sudoku, ο στόχος κάθε Kenken puzzle είναι να γεμίσει ένα πλέγμα με ψηφία από 1 έως 4 για ένα πλέγμα 4 × 4, από 1 έως 5 για ένα 5 × 5, κτλ έτσι ώστε κανένα ψηφίο να μην εμφανίζεται περισσότερες από μία φορές σε οποιαδήποτε σειρά ή σε οποιαδήποτε στήλη. Το μέγεθος του πλέγματος θα είναι δηλωμένο σαν σταθερά στο πρόγραμμά σας.

Επιπλέων, ένα πλέγμα KenKen χωρίζεται σε ομάδες κελιών που ονομάζονται **"κλουβιά" (cages),** τα οποία έχουν τον περιορισμό ότι: *οι αριθμοί των κελιών σε κάθε κλουβί πρέπει να παράγουν έναν συγκεκριμένο αποτέλεσμα όταν συνδυάζονται χρησιμοποιώντας ένα δοσμένο μαθηματικό τελεστή.* Για αυτήν την εργασία, οι μαθηματικές *τελεστές* που πρέπει να υλοποιηθούν, είναι η πρόσθεση και ο πολλαπλασιασμός. Δυο κελιά θεωρούνται ότι μπορούν να ανήκουν στο ίδιο κλουβί, εάν έχουν κοινή 1 πλευρά (τα κελιά τα οποία έχουν εφαπτόμενες γωνίες -corners - δεν μπορούν να θεωρούνται μέλη του ίδιου κλουβιού

Για παράδειγμα, στο puzzle της εικόνας 1, οι αριθμοί των κελιών του πάνω αριστερού κλουβιού, πρέπει να έχουν γινόμενο 16.



Εικόνα 1 Kenken puzzle (αριστερά) και μια από τις ενδεχόμενα περισσότερες λύσεις του (δεξιά).

Ζητούμενα Άσκησης

Ζητούμενο αυτής της άσκησης είναι η κατασκευή ενός προγράμματος, `kenken.c`, στη γλώσσα προγραμματισμού C, το οποίο δοθέντος ενός πλέγματος Kenken puzzle, επιτρέπει στον χρήστη να βρει μια λύση. Το πρόγραμμα πρέπει να υλοποιεί τις ακόλουθες λειτουργίες:

- **Λειτουργία 1 (Ανάγνωση Αρχείου)**

  Ο χρήστης δίνει στην γραμμή εντολών το όνομα ενός αρχείου `inputfile.txt` που περιέχει ένα Kenken puzzle το οποίο διαβάζεται στην μνήμη. Εντάξετε στο πρόγραμμα σας μια συνάρτηση `readPuzzle(your_parameters)` (όπου `your_parameters` μπορεί να είναι όσες παράμετροι θεωρείτε αναγκαίες) η οποία θα διαβάζει ένα άλυτο Kenken puzzle από ένα αρχείο κειμένου σε μια δομή δεδομένων (π.χ., `short kenken[N][N];`).
  Ένα Kenken puzzle αποθηκεύεται ως ακολούθως: Η πρώτη γραμμή του αρχείου πρέπει να καθορίζει το μέγεθος του puzzle. Κάθε επόμενη γραμμή θα πρέπει να αναφέρεται σε ένα κλουβί. Το πρώτο στοιχείο σε κάθε γραμμή περιέχει τον περιορισμό (π.χ. "**16***") ακολουθούμενο από τις συντεταγμένες των κελιών του κλουβί. Για παράδειγμα, το παραπάνω puzzle της εικόνας 1 θα παρουσιαστεί σε ένα αρχείο ως:

```
4
16* (1,1) (1,2) (2,2)
7+ (1,3) (1,4) (2,3)
4+ (2,1) (3,1)
4 (2,4)
12* (3,2) (4,1) (4,2)
8* (3,3) (3,4)
3+ (4,3) (4,4)
```

Μπορείτε να ανακτήσετε περισσότερα αρχεία εισόδου από το συνοδευτικό αρχείο `as1-supplementary.zip` ή παράγοντας τα δικά σας αρχεία εισόδου (αλλά και τις λύσεις) με χρήση της ιστοσελίδας https://tirl.org/software/kenken/

Η αρχική παρουσίαση του Kenken puzzle της εικόνας 1 στην οθόνη πρέπει να είναι στην παρακάτω μορφή:

```
a=16* b=7+ c=4+ d=4+ e=12* f=8* g=3+
+------+------+------+------+
|  a   |  a   |  b   |  b   |
+------+------+------+------+
|  c   |  a   |  b   |  d   |
+------+------+------+------+
|  c   |  e   |  f   |  f   |
+------+------+------+------+
|  e   |  e   |  g   |  g   |
+------+------+------+------+
```

Εικόνα 2 Παρουσίαση Kenken puzzle

Όπως φαίνεται στην εικόνα 2, το κάθε κλουβί τα οποία αναπαρίσταται από ένα χαρακτήρα του αλφαβήτου. Στο πρώτο κλουβί (καθώς διαβάζονται οι γραμμές του αρχείου) δίνεται ο χαρακτήρας 'a', στο 2ο ο 'b', στο 3ο ο 'c' κλπ. Επίσης, στην κορυφή του puzzle εμφανίζονται οι τιμές που καθορίζουν τους περιορισμούς του Kenken puzzle.

Σημείωση: Το πρόγραμμα σας θα πρέπει να ορίζει μια σταθερά π.χ., `#define N 10` η οποία θα προσδιορίζει (πριν την μεταγλώττιση) το μέγιστο μέγεθος του Kenken puzzle που μπορεί να χειριστεί το πρόγραμμα σας. Κατά την διάρκεια της ανάγνωσης του αρχείου πρέπει να κάνετε τους κατάλληλους ελέγχους για την ορθότητα του μεγέθους και της δομής του αρχείου και να δώσετε τα κατάλληλα μηνύματα λάθους εάν αυτό απαιτείται. Υποθέστε επίσης ότι για προτακτικούς λόγους το puzzle δεν θα έχει παραπάνω από 26 κλουβιά (πληθικός αριθμός των πεζών χαρακτήρων του λατινικού αλφαβήτου).

- **Λειτουργία 2 (Διαδραστική Επίλυση Kenken puzzle)**

Εντάξετε στο πρόγραμμα σας μια συνάρτηση `play(your_parameters)` η οποία θα επιτρέπει σε ένα παίκτη να εισάγει επαναληπτικά κάποιες τιμές, βάση των κανονισμών του παιχνιδιού, μέχρι να επιλυθεί το Kenken. Πιο συγκεκριμένα, όταν ξεκινήσει η εκτέλεση του παιχνιδιού σας τότε ο χρήστης καλείται να εισάγει κάποια δεδομένα στην μορφή `"i,j=val"`. Κάθε φορά που ο χρήστης δίνει ένα αριθμό, το πλέγμα πρέπει να αλλάξει μορφή, π.χ., σε κάποια χρονική στιγμή αν θέλει να δώσει τιμή για το κελί στη $2^η$ γραμμή και $2^η$ στήλη, το πλέγμα πρέπει να εμφανιστεί ως παρακάτω (προσέξετε ότι οι τιμές των κελιών έχουν δεξιά στοίχιση):

```
Enter your command in the following format:
>i,j=val: for entering val at position (i,j)
>i,j=0  : for clearing cell (i,j)
>0,0=0  : for saving and ending the game
Notice: i,j,val numbering is from [1..4]
>2,2=2

Value inserted!
+------+------+------+------+
|  2a  |  4a  |   b  |   b  |
+------+------+------+------+
|  1c  |  2a  |   b  |  4d  |
+------+------+------+------+
|  3c  |   e  |   f  |   f  |
+------+------+------+------+
|   e  |   e  |   g  |   g  |
+------+------+------+------+
```

Το παιχνίδι θα πρέπει να υποστηρίζει τις εξής περιπτώσεις:

a. **Εισαγωγή Αριθμού (εντολή: i,j=val):** Αν ο χρήστης δώσει ως εντολή εισαγωγής 1,2=4 τότε δηλώνει ότι θέλει να τοποθετήσει τον αριθμό 4 στην 1η γραμμή και 2η στήλη (σημειώστε ότι για ένα 4x4 Kenken οι γραμμές αριθμούνται από το 1 έως το 4). Στη συνέχεια, γίνεται έλεγχος αν η τιμή που έδωσε ο χρήστης μπορεί να εισαχθεί στο συγκεκριμένο κελί βάσει των κανόνων του Kenken (υπόδειξη: μπορείτε να φτιάξετε μια ξεχωριστή συνάρτηση για να κάνει τον έλεγχο αυτό, την οποία θα καλείτε από την play()). Εάν η εισαγωγή ολοκληρώνει το παιχνίδι τότε ο χρήστης πρέπει να λαμβάνει το κατάλληλο μήνυμα και να εκτυπώνεται το επιλυμένο Kenken στην οθόνη. Εάν δεν ολοκληρώνεται το παιχνίδι τότε απλά ξανά-παρουσιάζεται το Kenken στην οθόνη και ζητείται από τον χρήστη να δώσει την επόμενη του εισαγωγή.

   Η συνάρτηση play() πρέπει να δίνει τα κατάλληλα μηνύματα λάθους εάν:

   - Ο χρήστης δίνει εντολή σε άλλη μορφή ή δίνει μια μη έγκυρη τιμή στο i,j,val.
   - Ο χρήστης ζητά να γίνει η τοποθέτηση σε κελί που περιέχει ήδη κάποιο άλλο αριθμό (είτε δοσμένο ή του χρήστη) ή σε θέση εκτός των ορίων του πίνακα Kenken.

   Υπόδειξη: Για να διαβάσετε τρεις αριθμούς στην μορφή που ζητείται μπορείτε να χρησιμοποιήσετε την ακόλουθη εντολή.

```
if (scanf("%d,%d=%d",&i,&j,&val) != 3) {
       while (getchar() != '\n') {};
       printf("wrong format of command\n");
}
```

   /* Η πιο πάνω εντολή διαβάζει 3 αριθμούς από τον χρήστη όπου ο πρώτος χωρίζεται από τον δεύτερο με κόμμα και ο τρίτος χωρίζεται από τον δεύτερο με ίσον. Αν η scanf επιστρέψει 3, η ανάγνωση ήταν επιτυχής και θα εκτελεστεί η εντολή που ακολουθεί το if . Στην αντίθετη περίπτωση διαβάζονται και αγνοούνται οι χαρακτήρες μέχρι την επόμενη αλλαγή γραμμής.*/

b. **Διαγραφή Αριθμού (εντολή: i,j=0):** Αν ο χρήστης δώσει ως εντολή εισαγωγής το **1,2=0** τότε δηλώνει ότι θέλει να διαγράψει το περιεχόμενο του κελιού (1,2).

c. **Διακοπή Παιχνιδιού (εντολή: 0,0=0):** Αν ο χρήστης δώσει ως εντολή εισαγωγής **0,0=0** τότε δηλώνει ότι θέλει να σταματήσει την εκτέλεση του παιχνιδιού. Σε αυτή την περίπτωση διακόπτεται το παιχνίδι και αποθηκεύεται ο πίνακας Kenken σε αρχείο όπως περιγράφεται στη λειτουργία 3.

Η σύνταξη κλήσης του προγράμματος για αυτήν την λειτουργία θα είναι:

```
./kenken -i inputFileName
```

όπου:

- `-i` – είναι το όρισμα που λέει στο πρόγραμμα ότι πρέπει να εκτελέσει τη Λειτουργία 2.
- `inputFileName` – είναι το όνομα του αρχείου εισόδου που περιέχει τα δεδομένα του Kenken puzzle.

- **Λειτουργία 3 (Αποθήκευση λύσης)**

Εντάξετε στο πρόγραμμα σας μια συνάρτηση `writePuzzle(your_parameters)` η οποία αποθηκεύει στο αρχείο `out-filename.txt` (όπου `filename.txt` το όνομα του αρχείου που περιείχε το αρχικό παιχνίδι) το Kenken puzzle που βρίσκεται στην μνήμη. Για παράδειγμα, για το puzzle της εικόνας 1, το αρχείο εξόδου πρέπει να περιέχει τους αριθμούς:

```
2a 4a 1b 3b
1c 2a 3b 4d
3c 1e 4f 2f
4e 3e 2g 1g
```

Αυτή τυγχάνει να είναι και μια λύση του puzzle της εικόνας 1.

- **Λειτουργία 4 (Αυτόματη Επίλυση Kenken puzzle)**

Για να λυθεί αυτόματα ένα Kenken puzzle, μπορούμε να χρησιμοποιήσουμε ένα αλγόριθμο ο οποίος παράγει όλους του συνδυασμούς τιμών για τα κελιά (περιοριστείτε για λόγους απόδοσης σε μικρές τιμές του N, π.χ., 5) που ικανοποιούν τους περιορισμούς του παιχνιδιού, δηλαδή κάθε γραμμή και κάθε στήλη φέρουν μια διακριτή ακέραια τιμή και επίσης να ικανοποιούνται οι περιορισμοί των κλουβιών όπως αναλύθηκαν νωρίτερα. Η πιο πάνω διαδικασία, η οποία είναι brute-force (τυφλή ή απληροφόρητη), είναι η απλούστερη και υπολογιστικά ακριβότερη μέθοδος για επίλυση τέτοιων προβλημάτων, ωστόσο επαρκεί για να αναδείξει τη χρησιμότητα των υπολογιστών για αυτόματη επίλυση προβλημάτων

Για αυτήν την περίπτωση, η σύνταξη κλήσης του προγράμματος για αυτήν την λειτουργία θα είναι:

```
$ ./kenken -s inputFileName
```

όπου:

- `-s` - είναι το όρισμα που λέει στο πρόγραμμα ότι πρέπει να εκτελέσει τη Λειτουργία 3.
- `inputFileName` - είναι το όνομα του αρχείου εισόδου που περιέχει τα δεδομένα του Kenken puzzle.

Παρακαλώ όπως χρονομετρήσετε τον χρόνο εκτέλεσης του solver σας με τον πιο κάτω τρόπο:

```
$ time ./kenken -s inputFileName

real   0m0.987s
user   0m0.974s (αυτό σε ένα Intel Core Duo θα πάρει σχεδόν 1 δευτερόλεπτο!)
sys    0m0.007s
```

Εάν έχετε χρόνο και διάθεση, προσπαθήσετε να σκεφτείτε τρόπους για να βελτιώσετε τον χρόνο εκτέλεσης του επιλυτή σας. Οι καλύτερες λύσεις ενδέχεται να πάρουν και μπόνους στη βαθμολογία.

III. Γενικές Οδηγίες

Το πρόγραμμα σας θα πρέπει να συμβαδίζει με το πρότυπο ISO C, να περιλαμβάνει εύστοχα και περιεκτικά σχόλια, να έχει καλή στοίχιση και το όνομα κάθε μεταβλητής, σταθεράς, ή συνάρτησης να είναι ενδεικτικό του ρόλου της. **Να χρησιμοποιήσετε το λογισμικό τεκμηρίωσης doxygen** έτσι ώστε να μπορούμε να μετατρέψουμε τα σχόλια του προγράμματός σας σε HTML αρχεία και να τα δούμε με ένα browser. Η συστηματική αντιμετώπιση της λύσης ενός προβλήματος περιλαμβάνει στο παρόν στάδιο τη διάσπαση του προβλήματος σε μικρότερα ανεξάρτητα προβλήματα που κατά κανόνα κωδικοποιούμε σε ξεχωριστές συναρτήσεις. Για αυτό τον λόγο σας καλούμε να κάνετε χρήση συναρτήσεων και άλλων τεχνικών δομημένου προγραμματισμού που διδαχτήκατε στο ΕΠΛ131. Επίσης, σας θυμίσουμε ότι κατά την διάρκεια της εκτέλεσης του προγράμματος σας αυτό θα πρέπει να δίνει τα κατάλληλα μηνύματα σε περίπτωση λάθους. Τέλος το πρόγραμμα σας θα πρέπει να μεταγλωττίζεται στις μηχανές του εργαστηρίου. **Παρακαλώ όπως μελετηθούν ξανά οι κανόνες υποβολής εργασιών όπως αυτοί ορίζονται στο συμβόλαιο του μαθήματος.** Επίσης να ακολουθήσετε τα πιο κάτω βήματα όταν υποβάλετε την άσκηση σας στο Moodle:

1. Δημιουργείστε ένα κατάλογο με το όνομά σας π.χ., PyrrosBratskas/ χωρίς να αφήνετε κενά στο όνομα του καταλόγου.
2. Βάλτε μέσα στον κατάλογο αυτό όλα τα αρχεία της εργασίας (κώδικας, doxygen configuration file, README.md) που πρέπει να υποβάλετε.
3. Συμπιέστε (zip) τον κατάλογο (και όχι τα αρχεία ξεχωριστά) χρησιμοποιώντας την εντολή *zip –c PyrrosBratskas.zip PyrrosBratskas/**
4. **Βεβαιωθείτε ότι κάνατε σωστά τα τρία προηγούμενα βήματα**
5. Ανεβάστε στο Moodle το συμπιεσμένο αρχείο π.χ., PyrrosBratskas.zip

IV. Κριτήρια αξιολόγησης

Λειτουργία 1 (Ανάγνωση Αρχείου)	10
Λειτουργία 2 (Διαδραστική επίλυση Kenken)	35
Λειτουργία 3 (Αποθήκευση Kenken):	10
Λειτουργία 4 (Αυτόματη επίλυση Kenken):	35
Γενική εικόνα (ευανάγνωστος κώδικας, σχολιασμός, κλπ.)	10
ΣΥΝΟΛΟ	**100**