Τμήμα Εφαρμοσμένης Πληροφορικής
ΔΙΑΔΙΚΑΣΤΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ
Εξάμηνο Α'
Φύλλο Ασκήσεων 6 – ΔΕΙΚΤΕΣ

Παρατηρήσεις:

1. *Τα δεδομένα εισόδου διαβάζονται με τη σειρά που δηλώνονται στις εκφωνήσεις. Για κάθε δεδομένο εισόδου να χρησιμοποιείτε προτρεπτικό μήνυμα που θα ενημερώνει τον χρήστη για την τιμή που αναμένεται.*
2. *Αντίστοιχα για τα δεδομένα εξόδου και όπου δεν υπάρχουν περαιτέρω διευκρινήσεις για τη μορφή τους, αυτά θα εμφανίζονται με ξεχωριστές εντολές* `printf("…\n")` *το καθένα και με τη σειρά που δηλώνονται στις εκφωνήσεις.*
3. *Τα αριθμητικά δεδομένα αναπαρίστανται πάντα από μεταβλητές ακέραιου τύπου (*`int` *ή* `long`*). Σε αντίθετη περίπτωση (μεταβλητές τύπου* `double`*) θα γίνονται οι απαραίτητες διευκρινήσεις.*
4. *Η εμφάνιση τιμών τύπου* `float` *και* `double` *θα γίνεται με την εξής μορφοποίηση:* `.1f` *και* `.11f` *αντίστοιχα, εκτός κι αν ορίζεται διαφορετικά στην άσκηση.*
5. *Χρήση του όρου «Επιστρέφει» μέσα σε εισαγωγικά: Στις ακόλουθες ασκήσεις όταν υπάρχει όρος «επιστρέφει», δεν σημαίνει κατά ανάγκη ότι οι τιμές που υπολογίζονται από την κληθείσα συνάρτηση θα επιστρέφονται με την εντολή return. Απαιτείται η κληθείσα συνάρτηση να υπολογίζει τις τιμές και να ενημερώνει κατάλληλα την καλούσα συνάρτηση (πχ με χρήση δεικτών).*

---

1. Σε ένα απλό αισθητήρα μιας συσκευής ο χρόνος μετράται σε δευτερόλεπτα από τα μεσάνυχτα. Έτσι για παράδειγμα όταν το ρολόι της συσκευής δείχνει 4812, η ώρα είναι 1:20:12. Γράψτε μια συνάρτηση `GetTime` με το ακόλουθο πρωτότυπο:

```
void GetTime(long SysSecs, int *hours,int *minutes,int *seconds);
```

που να δέχεται ένα `long` ο οποίος δηλώνει τα δευτερόλεπτα που αναφέρονται στο ρολόι της συσκευής και επιστρέφει την ώρα, τα λεπτά και τα δευτερόλεπτα που αντιστοιχούν στον κλασικό τρόπο μέτρησης του χρόνου της ημέρας.

Δοκιμάστε τη συνάρτησή σας γράφοντας ένα απλό κυρίως πρόγραμμα που να μπορεί να παράγει το ακόλουθο δείγμα εκτέλεσης:

```
Enter Device Secs: 4812
Time is 1:20:12
```

2. Να γραφεί ένα πρόγραμμα το οποίο διαβάζει μη-αρνητικούς ακέραιους αριθμούς (int) από τον χρήστη (δεν χρειάζεται έλεγχος) και θα τους αποθηκεύει σε ένα μονοδιάστατο πίνακα, μέγιστου μεγέθους 100. Το πρόγραμμα δέχεται αριθμούς από τον χρήστη μέχρι εκείνος να εισάγει την τιμή -1. Έπειτα υπολογίζει το μέγιστο και ελάχιστο στοιχείο του πίνακα και εμφανίζει (από το κυρίως πρόγραμμα, δηλαδή την συνάρτηση `main()`) κατάλληλο μήνυμα στην οθόνη όπως φαίνεται στο παράδειγμα εκτέλεσης. Το πρόγραμμά σας θα πρέπει να έχει:

- μια συνάρτηση **readData** η οποία διαβάζει τους αριθμούς από τον χρήστη, αποθηκεύει στον πίνακα (όλους τους αριθμούς εκτός από το -1), και επιστρέφει το πλήθος των τιμών που εισήγαγε ο χρήστης χωρίς να υπολογίζει την τιμή -1,
- μια συνάρτηση **findMinMax** η οποία εντοπίζει και ενημερώνει την καλούσα συνάρτηση συγχρόνως τόσο για τη μικρότερη όσο και για τη μεγαλύτερη τιμή ενός μονοδιάστατου πίνακα ακεραίων, δηλαδή «επιστρέφει» (δείτε παρατήρηση 5 στην αρχή του φυλλαδίου) δύο τιμές.

<u>Προσοχή! Τα ονόματα των συναρτήσεων θα πρέπει να είναι αυτά που αναγράφονται πιο πάνω.</u>

Παρακάτω δίνεται δείγμα εκτέλεσης:

```
Enter the elements of the array, one per line.
Use -1 to signal the end of the list.
? 67
? 78
? 75
? 70
? 71
? 80
```

```
? 69
? 86
? 65
? 54
? 76
? 78
? 70
? 68
? 77
? -1
The range of values is 54-86
```

3. Να γραφεί πρόγραμμα το οποίο θα διαβάζει τρεις ακέραιους αριθμούς και θα τους καταχωρεί στις μεταβλητές `A`, `B`, `C`, θα τους ταξινομεί κατά αύξουσα σειρά εναλλάσσοντας τις τιμές τους μεταξύ των μεταβλητών με τη βοήθεια της συνάρτησης `swap(x,y)`, έτσι ώστε `A<B<C`. Τέλος θα εμφανίζει τις ταξινομημένες τιμές των μεταβλητών `A`, `B` και `C`. Η συνάρτηση `swap(int *x, int *y)` θα εναλλάσσει τις τιμές 2 ακεραίων μεταβλητών `x` και `y` (η τιμή της `x` θα δοθεί στην μεταβλητή `y` και της `y` στην `x`).

4. Να γραφεί ένα πρόγραμμα που θα περιλαμβάνει δύο συναρτήσεις:

- Μία συνάρτηση στην οποία θα διαβάζονται από το πληκτρολόγιο ακέραιες τιμές σε ένα διδιάστατο πίνακα 10×20
- Μία συνάρτηση που θα δέχεται ένα πίνακα 10×20 και θα επιστρέφει την τιμή του μεγαλύτερου στοιχείου του πίνακα και τη θέση του (γραμμή και στήλη).

Οι δύο συναρτήσεις θα καλούνται από το κυρίως πρόγραμμα σειριακά.

5. Να γραφεί ένα πρόγραμμα που θα εκτελεί τις παρακάτω λειτουργίες με τη χρήση διαδικασιών ή συναρτήσεων

a. **inputArray**: Θα διαβάζει N ακέραιους αριθμούς και θα τους καταχωρεί στο μονοδιάστατο πίνακα. Μέγιστη διάσταση μονοδιάστατου πίνακα 100 ($N \leq 100$). Δεν απαιτείται έλεγχος.

b. **inputNumbers**: Θα διαβάζει 2 ακέραιους αριθμούς A, B όπου A<B

c. **performCaclulations**: Θα βρίσκει: (1) το πλήθος των αριθμών (που διάβασε στο ερώτημα **b**) που είναι μικρότεροι ή ίσοι του A, (2) το πλήθος των αριθμών που είναι μεγαλύτεροι ή ίσοι του B, και (3) το πλήθος των αριθμών που είναι μεγαλύτεροι του A και μικρότεροι του B.

d. **ShowResults:** Θα εμφανίζει τους N ακέραιους, τους αριθμούς A και B, και τις τιμές που προσδιόρισε στο ερώτημα c.

Παρακάτω δίνεται δείγμα εκτέλεσης:

```
Give n: 10
Give element [0]: 34
Give element [1]: 23
Give element [2]: 56
Give element [3]: 78
Give element [4]: 12
Give element [5]: 9
Give element [6]: 80
Give element [7]: 45
Give element [8]: 63
Give element [9]: 45
Give A: 30
Give B: 60

----The numbers of the array-----
34 23 56 78 12 9 80 45 63 45
A = 30
B = 60
Numbers smaller or equal to A: 3
Numbers bigger or equal to B: 3
Numbers bigger than A and smaller than B: 4
```

6. Να γραφεί ένα πρόγραμμα το οποίο εμφανίζει μέσους όρους σωματομετρικών στοιχείων μιας ομάδας Ν ατόμων (το Ν είναι "σταθερά" και καθορίζεται με αντίστοιχη εντολή `define`). Για την υποβολή της άσκησης μπορείτε να θέσετε το Ν ίσο με 5. Το *πρόγραμμα* θα εκτελεί τις παρακάτω λειτουργίες *με τη χρήση συναρτήσεων*:

a. Θα διαβάζει Ν τετράδες ακέραιων αριθμών (σωματομετρικά στοιχεία) και θα τους καταχωρεί σε διδιάστατο πίνακα (συνάρτηση `readData`). Κάθε τετράδα αναφέρεται στα στοιχεία ενός ατόμου, που είναι κατά σειρά τα εξής:

- το φύλο (0 αν είναι άνδρας, 1 αν είναι γυναίκα)
- το βάρος (σε κιλά, ακέραια τιμή)
- το ύψος (σε εκατοστά, ακέραια τιμή)
- η ηλικία (σε χρόνια, ακέραια τιμή)

b. Θα εμφανίζει το μέσο όρο του βάρους, του ύψους και της ηλικίας τόσο για τους άνδρες όσο και για τις γυναίκες. Ο υπολογισμός (και όχι η εμφάνιση) του μέσου όρου για κάθε ένα στοιχείο (ύψος, βάρος ηλικία) θα γίνεται με τη βοήθεια μιας (και μόνο) συνάρτησης void `FindMean`. Εκτός των άλλων παραμέτρων που θα ορίσετε στη συνάρτηση, αυτή θα έχει ως παράμετρο και *έναν αριθμοδείκτη (τιμές 1, 2, 3)*. Ο αριθμοδείκτης θα *δείχνει* το αντίστοιχο στοιχείο της τετράδας, του οποίου ο μέσος όρος υπολογίζεται στη συγκεκριμένη κλήση, δηλαδή τη κατάλληλη στήλη του διδιάστατου πίνακα. Για παράδειγμα, όταν ο αριθμοδείκτης είναι 1, τότε θα υπολογίζεται ο μέσος όρος του βάρους. Η συνάρτηση `FindMean` σε κάθε κλήση της θα «επιστρέφει» συγχρόνως τον αντίστοιχο μέσο όρο για τους άνδρες και τον αντίστοιχο μέσο όρο για τις γυναίκες, θα «επιστρέφει» δηλαδή 2 τιμές. (δείτε παρατήρηση 5 για τον όρο «επιστρέφει»).

Στη συνάρτηση `main()` θα καλούνται κατάλληλα οι παραπάνω συναρτήσεις/διαδικασίες προκειμένου να διαβαστούν τα δεδομένα και στη συνέχεια να εμφανιστούν οι μέσοι όροι του βάρους, του ύψους και της ηλικίας των ανδρών και των γυναικών. Οι μέσοι όροι θα εμφανίζονται με ακρίβεια ενός δεκαδικού ψηφίου.

Παράδειγμα εκτέλεσης (για Ν 5).

```
Dwse fylo: 0
Dwse baros: 90
Dwse ypsos: 180
Dwse ilikia: 40
-----
Dwse fylo: 1
Dwse baros: 65
Dwse ypsos: 160
Dwse ilikia: 42
-----
Dwse fylo: 1
Dwse baros: 93
Dwse ypsos: 160
Dwse ilikia: 35
-----
Dwse fylo: 0
Dwse baros: 65
Dwse ypsos: 160
Dwse ilikia: 25
-----
Dwse fylo: 1
Dwse baros: 45
Dwse ypsos: 170
Dwse ilikia: 37
-----
Mesos oros barous andrwn: 77.5
Mesos oros barous gynaikwn: 67.7

Mesos oros ypsous andrwn: 170.0
Mesos oros ypsous gynaikwn: 163.3

Mesos oros hlikias andrwn: 32.5
Mesos oros hlikias gynaikwn: 38.0
```

7. Μια αλυσίδα κινηματογράφων έχει θερινούς κινηματογράφους σε πόλεις της Ελλάδας και της Κύπρου (1 αίθουσα/πόλη) αν και σκέφτεται να επεκτείνει το δίκτυο κινηματογράφων και σε άλλες χώρες των Βαλκανίων. Η εταιρεία αυτή θέλει να αναπτύξει ένα πρόγραμμα για να καταγράφει τις εισπράξεις και να υπολογίζει και να εμφανίζει διάφορα απλά στατιστικά στοιχεία (ανά χώρα). Οι μηνιαίες εισπράξεις κάθε αίθουσας για ένα καλοκαίρι καταχωρούνται σε πίνακα δύο διαστάσεων (πόλεις, μήνες). Το πρόγραμμα θα εκτελεί τις παρακάτω 6 λειτουργίες με τη χρήση συναρτήσεων:

1. Συνάρτηση που θα διαβάζει τις μηνιαίες εισπράξεις κάθε πόλης για ένα καλοκαίρι και θα τις αποθηκεύει σε πίνακα 2 διαστάσεων. Το καλοκαίρι οι κινηματογράφοι λειτουργούν και τους 3 μήνες. Η συνάρτηση θα έχει παραμέτρους τις διαστάσεις του πίνακα, τον πίνακα και το αντίτιμο του εισιτηρίου. Θα διαβάζει τον αριθμό των εισιτηρίων κάθε πόλης για κάθε καλοκαιρινό μήνα και θα υπολογίζει την αντίστοιχη είσπραξη με βάση το καθορισμένο αντίτιμο του εισιτηρίου (δες παρακάτω για το αντίτιμο του εισιτηρίου).
2. Συνάρτηση AvgCities που θα ***υπολογίζει*** τη μέση μηνιαία είσπραξη για κάθε πόλη.
3. Συνάρτηση AvgMonths που θα ***υπολογίζει*** τη μέση μηνιαία είσπραξη για κάθε μήνα.
4. Συνάρτηση FindMin που θα ***υπολογίζει*** τη μικρότερη μηνιαία είσπραξη και σε ποια πόλη και σε ποιο μήνα πραγματοποιήθηκε.
5. Συνάρτηση Print που θα ***εμφανίζει*** τις εισπράξεις των κινηματογράφων για κάθε πόλη και κάθε μήνα.
6. Συνάρτηση print2Tab που θα ***εμφανίζει*** τη μέση μηνιαία είσπραξη για κάθε αίθουσα, τη μέση μηνιαία είσπραξη για κάθε μήνα, τη μικρότερη μηνιαία είσπραξη και σε ποια πόλη και σε ποιο μήνα πραγματοποιήθηκε.

Σας δίνεται ένα στιγμιότυπο εκτέλεσης του προγράμματος για 5 αίθουσες στην Ελλάδα και για 2 αίθουσες στην Κύπρο. Το αντίτιμο του εισιτηρίου στην Ελλάδα είναι 7.5 ευρώ και στην Κύπρο 8.5 ευρώ. Η εμφάνιση των ποσών των εισπράξεων θα γίνεται σε εύρος 6 θέσεων εκ των οποίων οι 2 τα δεκαδικά ψηφία.

```
Tickets in Greece
Give tickets: [0][0]: 15
Give tickets: [0][1]: 13
Give tickets: [0][2]: 24
Give tickets: [1][0]: 27
Give tickets: [1][1]: 28
Give tickets: [1][2]: 31
Give tickets: [2][0]: 41
Give tickets: [2][1]: 32
Give tickets: [2][2]: 33
Give tickets: [3][0]: 36
Give tickets: [3][1]: 19
Give tickets: [3][2]: 25
Give tickets: [4][0]: 27
Give tickets: [4][1]: 38
Give tickets: [4][2]: 45
Tickets in Cyprus
Give tickets: [0][0]: 36
Give tickets: [0][1]: 38
Give tickets: [0][2]: 48
Give tickets: [1][0]: 45
Give tickets: [1][1]: 29
Give tickets: [1][2]: 31

Cinema revenues in Greece
City 0:   112.50  97.50 180.00
City 1:   202.50 210.00 232.50
City 2:   307.50 240.00 247.50
City 3:   270.00 142.50 187.50
City 4:   202.50 285.00 337.50

Statistics in Greece
```

```
Cities average revenues
0: 130.00
1: 215.00
2: 265.00
3: 200.00
4: 275.00
Months average revenues
Month 0: 219.00
Month 1: 195.00
Month 2: 237.00
Min revenue:  97.50
City: 0
Month: 1

Cinema revenues in Cyprus
City 0:   306.00 323.00 408.00
City 1:   382.50 246.50 263.50

Statistics in Cyprus
Cities average revenues
0: 345.67
1: 297.50
Months average revenues
Month 0: 344.25
Month 1: 284.75
Month 2: 335.75
Min revenue: 246.50
City: 1
Month: 1
```

8. Να γίνει πρόγραμμα το οποίο θα δέχεται τους βαθμούς μιας τάξης 30 μαθητών στο μάθημα της χημείας και στη συνέχεια θα εμφανίζει τον βαθμό που παρατηρήθηκε τις περισσότερες φορές. Θεωρούμε ότι οι βαθμοί είναι ακέραιοι από 1 έως 20.

Το πρόγραμμα θα εκτελεί τις παρακάτω λειτουργίες με τη χρήση των εξής διαδικασιών ή συναρτήσεων, οι οποίες θα πρέπει να είναι συνολικά 5 σε αριθμό:

- **a.** Θα διαβάζει τις βαθμολογίες συγκεκριμένου πλήθους μαθητών και θα τις αποθηκεύει σε πίνακα. Παράμετροι συνάρτησης το μέγεθος του πίνακα και ο πίνακας.
- **b.** Θα υπολογίζει τη συχνότητα εμφάνισης του κάθε βαθμού. Η συνάρτηση θα δέχεται τον πίνακα με τις βαθμολογίες των μαθητών και θα ενημερώνει τον πίνακα με τις συχνότητες.
- **c.** Θα δέχεται τον πίνακα συχνοτήτων των βαθμών της κλίμακας 1..20 και θα υπολογίζει και θα «επιστρέφει» (δείτε παρατήρηση 5-αρχή του φυλλαδίου) ποιος βαθμός παρατηρήθηκε τις περισσότερες φορές (μεγαλύτερη συχνότητα) και ποιος βαθμός την παρουσίασε.
- **d.** Θα δέχεται τον πίνακα συχνοτήτων των βαθμών της κλίμακας 1..20 και θα υπολογίζει και θα «επιστρέφει» ποιος βαθμός παρατηρήθηκε τις λιγότερες φορές (μικρότερη συχνότητα) και ποιος βαθμός την παρουσίασε.
- **e.** Θα εμφανίζει τις συχνότητες των βαθμών, το βαθμό που παρατηρήθηκε τις περισσότερες φορές και πόσες φορές παρατηρήθηκε, το βαθμό που παρατηρήθηκε τις λιγότερες φορές και πόσες φορές παρατηρήθηκε.

Η υλοποίηση των συναρτήσεων θα πρέπει να συμμορφώνεται με το πρότυπο C99 όσον αφορά τις παραμέτρους πίνακα. Οι συναρτήσεις όπου "επιστρέφουν" περισσότερες από μια τιμές αυτό θα πρέπει να γίνεται με πέρασμα παραμέτρων με αναφορά (δείκτες).

Δίνεται ένα στιγμιότυπο εκτέλεσης:

```
Dwse bathmo 1: 1
Dwse bathmo 2: 2
Dwse bathmo 3: 3
Dwse bathmo 4: 4
Dwse bathmo 5: 5
Dwse bathmo 6: 5
```

```
Dwse bathmo 7: 6
Dwse bathmo 8: 7
Dwse bathmo 9: 8
Dwse bathmo 10: 9
Dwse bathmo 11: 10
Dwse bathmo 12: 11
Dwse bathmo 13: 12
Dwse bathmo 14: 13
Dwse bathmo 15: 14
Dwse bathmo 16: 15
Dwse bathmo 17: 16
Dwse bathmo 18: 16
Dwse bathmo 19: 17
Dwse bathmo 20: 18
Dwse bathmo 21: 19
Dwse bathmo 22: 4
Dwse bathmo 23: 5
Dwse bathmo 24: 5
Dwse bathmo 25: 6
Dwse bathmo 26: 7
Dwse bathmo 27: 17
Dwse bathmo 28: 15
Dwse bathmo 29: 11
Dwse bathmo 30: 10
Syxnotites bathmwn
1: 1
2: 1
3: 1
4: 2
5: 4
6: 2
7: 2
8: 1
9: 1
10: 2
11: 2
12: 1
13: 1
14: 1
15: 2
16: 2
17: 2
18: 1
19: 1
20: 0
Bathmos me th megalyteri syxnotita: 5
Syxnotita: 4
Bathmos me th mikroteri syxnotita: 20
Syxnotita: 0
```

9. Μια εταιρεία εμπορεύεται 5 προϊόντα αξίας 250, 150, 320, 210 και 920 ευρώ αντίστοιχα. Η πώληση των παραπάνω προϊόντων γίνεται μέσω 4 πωλητών. Ο παρακάτω πίνακας δίνει τις πωλήσεις που έγιναν μέσα σε μια βδομάδα:

Α/Α Πωλητή	Προϊόν 0	Προϊόν 1	Προϊόν 2	Προϊόν 3	Προϊόν 4
0	10	4	5	6	7
1	7	0	12	1	3
2	4	19	5	0	8
3	3	2	1	5	6

Αν ο πίνακας πωλήσεων είναι δεδομένος και σταθερός, και οι τιμές των προϊόντων είναι αποθηκευμένες σε σταθερό πίνακα κατάλληλης διάστασης, να γραφεί πρόγραμμα που θα:

- υπολογίζει και θα αποθηκεύει στον πίνακα `salesPerson[]` το συνολικό ποσό είσπραξης (`int`) για κάθε πωλητή, μέσω μιας συνάρτησης `calculateSales()`
- υπολογίζει και θα αποθηκεύει στον πίνακα `productSale[]` τις ποσότητες (`int`) που πουλήθηκαν από κάθε προϊόν, μέσω μιας συνάρτησης `calculateProductSales()`
- θα τυπώνει τις συνολικές πωλήσεις κάθε πωλητή και τις συνολικές πωλήσεις κάθε προϊόντος,
- Τον πωλητή με τις μεγαλύτερες (σε έσοδα) πωλήσεις και τα έσοδα του και το προϊόν με τις περισσότερες πωλήσεις και πόσα τεμάχια πούλησε.

Για τους παραπάνω υπολογισμούς να υλοποιήσετε

- μια συνάρτηση `maxArray()` η οποία θα δέχεται ένα μονοδιάστατο πίνακα, το μέγεθός του (αριθμό στοιχείων) και θα επιστρέφει το μέγιστό του στοιχείο (`int`) και την θέση του στον πίνακα (`int`),
- μια συνάρτηση `printArray()` η οποία δέχεται ένα πίνακα, την διάσταση του και θα τυπώνει τον πίνακα στην οθόνη σε δύο στήλες, όπου στην πρώτη στήλη θα δίνεται η θέση του στοιχείου στον πίνακα και στη δεύτερη η τιμή του.

Παρακάτω δίνεται παράδειγμα εκτέλεσης του προγράμματος.

```
SalesMan-Sales
   0     12400
   1     8560
   2     12810
   3     7940
Best SalesMan was 2 with 12810 sales
Product-Items
   0     24
   1     25
   2     23
   3     12
   4     24
Best Product was 1 with 25 items
```

10. Σας δίνεται ο σκελετός του προγράμματος **a10f6.c**, όπου δηλώνεται το πρωτότυπο και η υλοποίηση των συναρτήσεων `ReadData` και `findMax`. Η περιγραφή των συναρτήσεων που σας δίνονται είναι η ακόλουθη:

Συνάρτηση: **`void ReadData(int r, int c, double pin[r][c]);`**

- Η συνάρτηση `ReadData()` είναι μια void συνάρτηση (διαδικασία) και χρησιμοποιείται για να διαβάζονται πραγματικοί αριθμοί (double) και να καταχωρούνται σε ένα διδιάστατο πίνακα (τυπική παράμετρος pin) του οποίου το τρέχον μέγεθος είναι `r` γραμμές και `c` στήλες.
- Είναι μια συνάρτηση που μπορεί να χρησιμοποιηθεί σ' οποιοδήποτε πρόγραμμα που απαιτεί το διάβασμα πραγματικών αριθμών και οι οποίοι πρέπει να αποθηκεύονται σε διδάστατο πίνακα
- Εκεί απ' όπου θα κληθεί η συνάρτηση `ReadData` (ο καλών την συνάρτηση) θα πρέπει να δηλωθεί ο πίνακας με τις *κατάλληλες διαστάσεις* και το τρέχον πλήθος γραμμών και στηλών, τα οποία δέχεται ως ορίσματα.

Συνάρτηση: **`double findMax(int r, int c, double pin[r][c]);`**

- Η συνάρτηση `findMax()` είναι μια συνάρτηση που χρησιμοποιείται για να προσδιορισθεί το μέγιστο στοιχείο ενός διδιάστατου πίνακα.

- Δέχεται τον διδιάστατο πίνακα (τυπική παράμετρος `pin`), το τρέχον πλήθος γραμμών και στηλών (τυπικές παράμετροι `r` και `c` αντίστοιχα) και επιστρέφει το μέγιστο στοιχείο του διδιάστατου πίνακα.
- Εκεί απ' όπου θα κληθεί η συνάρτηση `findMax` (ο καλών την συνάρτηση) θα πρέπει να δηλωθεί ο πίνακας με τις κατάλληλες διαστάσεις και το τρέχον πλήθος γραμμών και στηλών, τα οποία δέχεται ως ορίσματα.

Χρησιμοποιείστε κατάλληλα τις παραπάνω συναρτήσεις για να λύσετε τα παρακάτω προβλήματα.

α) Ζητείται να υλοποιήσετε ένα πρόγραμμα που θα χρησιμοποιηθεί για τη βαθμολογία των μαθητών των ελληνικών σχολείων. Σας δίνεται ότι τα σχολεία στην Ελλάδα έχουν μέχρι το πολύ 21 τμήματα και κάθε τμήμα μέχρι το πολύ 15 μαθητές. Να υλοποιήσετε ένα πρόγραμμα που:

1. θα διαβάζει τον αριθμό των τμημάτων ενός συγκεκριμένου σχολείου και τον αριθμό των μαθητών ανά τμήμα. Όλα τα τμήματα έχουν τον ίδιο αριθμό μαθητών.
2. θα διαβάζει τους βαθμούς των μαθητών για κάθε τμήμα του σχολείου και θα τους αποθηκεύει σε διδιάστατο πίνακα. Σε κάθε γραμμή του πίνακα καταχωρούνται οι βαθμοί των μαθητών του ίδιου τμήματος.
3. θα υπολογίζει το μεγαλύτερο βαθμό μεταξύ όλων των μαθητών του σχολείου.
4. θα εμφανίζει το μεγαλύτερο βαθμό

τα ερωτήματα 2 και 3 θα γίνουν με τη βοήθεια των συναρτήσεων `ReadData` και `findMax` (που σας δίνονται) ενώ το 1 και 4 από τη συνάρτηση main χωρίς συναρτήσεις.

Ονομάστε το αρχείο a10af6.c

β) Στη συνέχεια σας ζητούν να επεκτείνετε το παραπάνω πρόγραμμα ώστε να υπολογίζει τα εξής:

5. το μεγαλύτερο βαθμό, καθώς και σε ποιο τμήμα σημειώθηκε και από ποιο μαθητή
6. για κάθε τμήμα το μεγαλύτερο βαθμό που σημείωσε μαθητής

- Για το 5. ερώτημα γράψετε μια νέα συνάρτηση την `findMaxRowCol` η οποία θα επιστρέφει το μεγαλύτερο βαθμό (μεγαλύτερη τιμή πίνακα), από ποιο μαθητή (στήλη) και σε ποιο τμήμα (γραμμή σημειώθηκε). (Υπόδειξη: χρησιμοποιήστε και τροποποιήστε κατάλληλα τον κώδικα της συνάρτησης `findMax`)
- Για το 6. ερώτημα γράψετε μια νέα συνάρτηση την `findMaxRow`, η οποία θα επιστρέφει μια λίστα με το μεγαλύτερο βαθμό κάθε τμήματος (Υπόδειξη: χρησιμοποιήστε και τροποποιήστε κατάλληλα τον κώδικα της συνάρτησης `findMax`).

Δηλώστε το πρωτότυπο, την υλοποίηση των συναρτήσεων `findMaxRow` και `findMaxRowCol` και να τις καλέσετε από τη main για να υπολογίζουν τα παραπάνω. Στη συνέχεια:

7. εμφανίστε το μεγαλύτερο βαθμό, καθώς και σε ποιο τμήμα σημειώθηκε και από ποιο μαθητή
8. λίστα με το μεγαλύτερο βαθμό κάθε τμήματος

Η εμφάνιση της λίστας με το μεγαλύτερο βαθμό κάθε τμήματος θα γίνεται με τη βοήθεια συνάρτησης ενώ η εμφάνιση του μεγαλύτερου βαθμού, ποιος μαθητής τον πέτυχε και σε ποιο τμήμα θα γίνεται στη main χωρίς χρήση συνάρτησης.

Ονομάστε το αρχείο a10bf6.c

11. Να γραφεί ένα πρόγραμμα το οποίο, με τη χρήση συναρτήσεων, θα εκτελεί τις παρακάτω λειτουργίες:

α. Θα διαβάζει τις μέσες ημερήσιες θερμοκρασίες ενός μήνα 30 ημερών και θα τις αποθηκεύει σε πίνακα με μία συνάρτηση `read_temperatures`, με κατάλληλα ορίσματα, (δηλαδή η συνάρτηση θα διαβάζει μια μέση θερμοκρασία για κάθε μέρα του μήνα).

β. Θα υπολογίζει την μέγιστη και ελάχιστη ημερήσια θερμοκρασία καθώς και τον μέσο όρο των θερμοκρασιών, χρησιμοποιώντας μια συνάρτηση find_max_min_average

γ. Το κυρίως πρόγραμμα (συνάρτηση main()) θα εμφανίζει τα αποτελέσματα που υπολογίζονται κατά το βήμα (β).

```
Dwse tis thermokrasies toy mina
Dwse ti thermokrasia 0: 27
Dwse ti thermokrasia 1: 22
Dwse ti thermokrasia 2: 30
Dwse ti thermokrasia 3: 18
Dwse ti thermokrasia 4: 22
Dwse ti thermokrasia 5: 23
```

. . .

```
Dwse ti thermokrasia 25: 19
Dwse ti thermokrasia 26: 22
Dwse ti thermokrasia 27: 17
Dwse ti thermokrasia 28: 25
Dwse ti thermokrasia 29: 23
H megisti thermokrasia einai 33
H elaxisti thermokrasia einai 17
O mesos oros einai 25.03
Press any key to continue . . .
```

12. Σας δίνεται το παρακάτω πρόγραμμα σε C (a12f6.c) που διαχειρίζεται βαθμολογίες αθλητών (στην κλίμακα 0..10) στα τρία αγωνίσματα του τρίαθλου (Κολύμβηση, Ποδήλατο, Τρέξιμο).

Ζητείται να ξαναγράψετε τον πηγαίο κώδικα του προγράμματος αξιοποιώντας συναρτήσεις για τα τμήματα κώδικα που επαναλαμβάνονται, διατηρώντας τη λειτουργικότητα του αρχικού προγράμματος αναλλοίωτη.

Το νέο πρόγραμμα θα πρέπει να περιλαμβάνει δυο επιπλέον συναρτήσεις

Α) ypologismosEpityxontwn: που θα υπολογίζει το μέσο όρο της βαθμολογίας του κάθε αθλητή στα τρία αθλήματα και θα εξάγει τους αθλητές που έχουν μέσο όρο βαθμολογίας πάνω από 5 (επιτυχόντες αθλητές). Η συνάρτηση θα πρέπει να επιστρέφει για κάθε <u>επιτυχόντα</u> αθλητή τον τριψήφιο κωδικό του και τον μέσο όρο της βαθμολογίας του.

Β) print: θα εμφανίζει τους κωδικούς των επιτυχόντων αθλητών και τους αντίστοιχους μέσους όρους της βαθμολογίας τους (μορφή εμφάνισης: `Kwdikos Athliti %d, Mesos oros vathmologias %.1f\n`).

```
#include <stdio.h>

#define ATHLITES 8

int main() {

    //Πίνακας που περιλαμβάνει τους τριψήφιους κωδικούς αθλητών
    int kwdikoiAthlitwn[ATHLITES] = {115, 136, 187, 254, 281, 290, 301, 314};

    //Πίνακες που περιλαμβάνουν τη βαθμολογία σε 3 αγωνίσματα
    float vathmologiaKolymbi[ATHLITES];
    float vathmologiaPodilato[ATHLITES];
    float vathmologiaTreximo[ATHLITES];

    int i;
    float maxKolymbi, maxPodilato, maxTreximo;
    float avgKolymbi, avgPodilato, avgTreximo;

    //Ανάγνωση βαθμολογίας κάθε αθλήματος από το χρήστη
    printf("EISAGWGI VATHMOLOGIAS ATHLIMATOS 1 - Kolymbi\n");
    for(i=0; i<ATHLITES; i++) {
        printf("Eisagete ti vathmologia tou athliti %d: ", i);
        scanf("%f", &vathmologiaKolymbi[i]);
    }

    printf("EISAGWGI VATHMOLOGIAS ATHLIMATOS 2 - Podilato\n");
    for(i=0; i<ATHLITES; i++) {
        printf("Eisagete ti vathmologia tou athliti %d: ", i);
        scanf("%f", &vathmologiaPodilato[i]);
    }

    printf("EISAGWGI VATHMOLOGIAS ATHLIMATOS 3 - Treximo\n");
    for(i=0; i<ATHLITES; i++) {
        printf("Eisagete ti vathmologia tou athliti %d: ", i);
        scanf("%f", &vathmologiaTreximo[i]);
    }

    //Υπολογισμός μέγιστης βαθμολογίας κάθε αθλήματος
    maxKolymbi = vathmologiaKolymbi[0];
    for(i=1; i<ATHLITES; i++)
       if(vathmologiaKolymbi[i] > maxKolymbi)
          maxKolymbi = vathmologiaKolymbi[i];
```

```
    maxPodilato = vathmologiaPodilato[0];
    for(i=1; i<ATHLITES; i++)
       if(vathmologiaPodilato[i] > maxPodilato)
          maxPodilato = vathmologiaPodilato[i];

    maxTreximo = vathmologiaTreximo[0];
    for(i=1; i<ATHLITES; i++)
       if(vathmologiaTreximo[i] > maxTreximo)
          maxTreximo = vathmologiaTreximo[i];

    printf("Megistes vathmologies: %.1f (Kol) %.1f (Pod) %.1f (Trex) \n",
            maxKolymbi, maxPodilato, maxTreximo);

    //Υπολογισμός μέσου όρου βαθμολογίας κάθε αθλήματος
    avgKolymbi = 0;
    for(i=0; i<ATHLITES; i++)
       avgKolymbi += vathmologiaKolymbi[i];
    avgKolymbi /= ATHLITES;

    avgPodilato = 0;
    for(i=0; i<ATHLITES; i++)
       avgPodilato += vathmologiaPodilato[i];
    avgPodilato /= ATHLITES;

    avgTreximo = 0;
    for(i=0; i<ATHLITES; i++)
       avgTreximo += vathmologiaTreximo[i];
    avgTreximo /= ATHLITES;

    printf("Mesoi oroi: %.1f (Kol) %.1f (Pod) %.1f (Trex) \n",
            avgKolymbi, avgPodilato, avgTreximo);

    return 0;
}
```

13. Να γραφεί μια συνάρτηση `decompose` τύπου **void,** η οποία θα δέχεται ως όρισμα έναν ακέραιο (`long`) και θα ενημερώνει την καλούσα συνάρτηση («επιστρέφει» - δείτε την παρατήρηση 5 στην αρχή του φυλλαδίου) για τα ακόλουθα:

- το πλήθος των ψηφίων του,
- τον μέσο όρο των ψηφίων του,
- το μέγιστο ψηφίο του.

Να υλοποιήσετε ένα πρόγραμμα το οποίο στη συνάρτηση `main()` θα ζητά από τον χρήστη ένα αριθμό (μη-αρνητικός ακέραιος-δεν απαιτείται έλεγχος), και χρησιμοποιεί την παραπάνω συνάρτηση για να εμφανίσει (*από τη συνάρτηση* `main()`) τα παραπάνω στοιχεία του αριθμού. Ο μέσος όρος θα εμφανίζεται με ακρίβεια 3 δεκαδικών ψηφίων. Παραδείγματα εκτέλεσης για διαφορετικές περιπτώσεις αριθμών ακολουθούν παρακάτω:

Παράδειγμα εκτέλεσης 1:

```
Please insert number:15
Digits: 2
Average: 3.000
Max: 5
```

Παράδειγμα εκτέλεσης 2:

```
Please insert number:1453
Digits: 4
Average: 3.250
Max: 5
```

Παράδειγμα εκτέλεσης 3:

```
Please insert number:1
Digits: 1
```

```
Average: 1.000
Max: 1
```

Παράδειγμα εκτέλεσης 4:

```
Please insert number:165878
Digits: 6
Average: 5.833
Max: 8
```

14. Ένας μαθητής θεωρείται ότι απέτυχε όταν ο μέσος όρος της βαθμολογίας του σε όλα τα μαθήματα είναι μικρότερος του 9.5, ενώ θεωρείται ότι αρίστευσε όταν ο μέσος όρος του είναι μεγαλύτερος ή ίσος του 18.5. Να γραφεί πρόγραμμα που θα υλοποιεί τις παρακάτω 4 συναρτήσεις:

1. Συνάρτηση που θα παράγει τυχαίους βαθμούς στην κλίμακα 0..20 για συγκεκριμένο πλήθος φοιτητών και μαθημάτων και θα τους καταχωρεί σε κατάλληλο πίνακα. Παράμετροι συνάρτησης διαστάσεις του πίνακα και ο πίνακας.
2. Συνάρτηση που θα δέχεται για συγκεκριμένο πλήθος φοιτητών και μαθημάτων τους βαθμούς τους και θα υπολογίζει το μέσο όρο της βαθμολογίας κάθε μαθητή και θα τον καταχωρεί σε κατάλληλο πίνακα.
3. Συνάρτηση που θα εμφανίζει για κάθε μαθητή τις βαθμολογίες του καθώς και τον αντίστοιχο μέσο όρο του (δες στιγμιότυπο εκτέλεσης).
4. Συνάρτηση που θα υπολογίζει και θα «επιστρέφει» (δείτε παρατήρηση 5 στην αρχή του φυλλαδίου) το πλήθος και το ποσοστό των μαθητών που απέτυχαν καθώς και το πλήθος και το ποσοστό των μαθητών που αρίστευσαν.

Η υλοποίηση των συναρτήσεων θα πρέπει να συμμορφώνεται με το πρότυπο C99 όσον αφορά τις παραμέτρους πίνακα. Οι συναρτήσεις όπου “επιστρέφουν" περισσότερες από μια τιμές αυτό θα πρέπει να γίνεται με πέρασμα παραμέτρων με αναφορά (δείκτες).

Στο κυρίως πρόγραμμα θα καλούνται οι παραπάνω συναρτήσεις και η εμφάνιση αυτών που υπολογίζει η συνάρτηση 4 θα γίνεται στο κυρίως πρόγραμμα. Θεωρείστε ότι έχετε 30 μαθητές και 3 μαθήματα.

Η εμφάνιση των πραγματικών αριθμών θα γίνεται σε εύρος 5 θέσεων με 2 δεκαδικά ψηφία.

Δίνεται ένα στιγμιότυπο εκτέλεσης:

```
 1:  1 13  4 :  6.00
 2:  6  1 14 :  7.00
 3: 12 12  7 : 10.33
 4: 15 11 12 : 12.67
 5:  3 10  4 :  5.67
 6:  2  2 10 :  4.67
 7: 10 18 18 : 15.33
 8: 19 12 11 : 14.00
 9:  7  5 12 :  8.00
10: 19  9 20 : 16.00
11:  9 19  8 : 12.00
12: 18 10  5 : 11.00
13: 17  4  4 :  8.33
14: 17  2  0 :  6.33
15: 11  6 19 : 12.00
16: 10  3 17 : 10.00
17: 12 12  8 : 10.67
18:  3 17 16 : 12.00
19:  0 16  2 :  6.00
20: 11  3 17 : 10.33
21: 11  1 19 : 10.33
22:  2  4  5 :  3.67
23:  3 17 16 : 12.00
24: 13 14  3 : 10.00
25:  9  4  4 :  5.67
26:  7  9 20 : 12.00
27: 14 13 11 : 12.67
28: 10 19  7 : 12.00
29:  1 18 20 : 13.00
```

```
30:  1 13 10 :  8.00
Apotyxontes: 11
Pososto apotyxonton: 36.00%
Aristouxoi: 0
Pososto aristouxwn:  0.00%
```

15. Να γραφεί πρόγραμμα που θα περιλαμβάνει τις εξής συναρτήσεις:

1. θα γεμίζει έναν πίνακα a μεγέθους MXN με τυχαίους αριθμούς από το 0 έως το 20 χρησιμοποιώντας την συνάρτηση rand() της βιβλιοθήκης <stdlib.h> (συνάρτηση populate.)
2. θα εμφανίζει τον πίνακα a στην οθόνη (συνάρτηση print_2d),
3. θα βρίσκει και θα επιστρέφει το μεγαλύτερο και το μικρότερο στοιχείο μιας συγκεκριμένης γραμμής του πίνακα. Η συνάρτηση θα δέχεται ως παράμετρο το μέγεθος της γραμμής του πίνακα (πλήθος στηλών) και τη γραμμή του πίνακα (συνάρτηση FindMaxMinRow)
   Η main θα εκτελεί τα εξής: α) θα καλεί τη συνάρτηση populate για πίνακα 5X4 β) θα εμφανίζει τα στοιχεία του πίνακα γ) θα δέχεται επαναληπτικά τον αύξοντα αριθμό μιας γραμμής του πίνακα a για να βρει το μεγαλύτερο και μικρότερο στοιχείο της συγκεκριμένης γραμμής, με τη χρήση της FindMaxMinRow δ) θα εμφανίζει το μεγαλύτερο και μικρότερο στοιχείο της συγκεκριμένης γραμμής. Η επανάληψη θα σταματάει αν δοθεί αριθμός εκτός των ορίων των γραμμών του πίνακα (αριθμός μεταξύ [0, M-1].
   Προσοχή: *πριν τη χρήση της rand() θα πρέπει να καλέσετε την srand(30) ώστε σε κάθε εκτέλεση του προγράμματος σας να παράγονται οι ίδιοι ψευδοτυχαίοι αριθμοί.*
   Δίνεται στιγμιότυπο εκτέλεσης

```
 10 13  2 17
  7  5 12  7
 19 14  9  1
 16  6  9 15
  4 20 13 15

Give line number 0
17, 2
Give line number 3
16, 6
Give line number 1
12, 5
Give line number 2
19, 1
Give line number 4
20, 4
Give line number -1
```

16. Δίνεται ο παρακάτω πίνακας ο οποίος περιέχει σωματομετρικά στοιχεία μαθητών

1, 40, 120, 12
0, 45, 110, 13
0, 50, 140, 14
1, 40, 120, 14
1, 45, 135, 15

Κάθε τετράδα αναφέρεται στα στοιχεία ενός μαθητή, που είναι κατά σειρά τα εξής:
το φύλο (0 αν είναι αγόρι, 1 αν είναι κορίτσι)
το βάρος (σε κιλά, ακέραια τιμή)
το ύψος (σε εκατοστά, ακέραια τιμή)
η ηλικία (σε χρόνια, ακέραια τιμή)

Να γραφεί πρόγραμμα το οποίο θα βρίσκει το μεγαλύτερο βάρος, ύψος και ηλικία τόσο για τα αγόρια όσο και για τα κορίτσια. Ο υπολογισμός (και όχι η εμφάνιση) του μεγαλύτερου (βάρος, ύψος, ηλικία) θα γίνεται με τη βοήθεια της *ίδιας συνάρτησης* void `FindMax` (δηλαδή θα καλείται η `FindMax` για να βρει κάθε φορά ανάλογα με την παράμετρο που δίνεται (δείτε παρακάτω), είτε το μεγαλύτερο βάρος, ή το μεγαλύτερο ύψος ή τη μεγαλύτερη ηλικία. Εκτός των άλλων παραμέτρων που θα ορίσετε στη συνάρτηση, θα υπάρχει επιπλέον ως παράμετρος και *ένας αριθμοδείκτης (τιμές 1, 2, 3)*. Ο αριθμοδείκτης θα *δείχνει* το

αντίστοιχο στοιχείο της τετράδας, του οποίου η μεγαλύτερη τιμή υπολογίζεται στη συγκεκριμένη κλήση, δηλαδή τη κατάλληλη στήλη του διδιάστατου πίνακα. Για παράδειγμα, όταν ο αριθμοδείκτης είναι 1, τότε θα υπολογίζεται το μεγαλύτερο βάρος, όταν ο αριθμοδείκτης είναι 2, τότε θα υπολογίζεται το μεγαλύτερο ύψος, κτλ. Η συνάρτηση `FindMax` σε κάθε κλήση της θα «επιστρέφει» 2 τιμές την μεγαλύτερη τιμή για τα αγόρια και την μεγαλύτερη τιμή για τα κορίτσια (δείτε παρατήρηση 5 στην αρχή του φυλλαδίου για τον όρο «επιστρέφει»).

Στη συνάρτηση `main()` θα καλείται κατάλληλα (πολλές φορές) η συνάρτηση `FindMax` προκειμένου να υπολογιστούν και στη συνέχεια να εμφανιστούν κάθε φορά οι μεγαλύτερες τιμές του βάρους, του ύψους και της ηλικίας των αγοριών και των κοριτσιών. Οι τιμές θα εμφανίζονται με ακρίβεια ενός δεκαδικού ψηφίου.

Στιγμιότυπο εκτέλεσης .

```
Max weight boy: 50.0
Max weight girl: 45.0
Max height boy: 140.0
Max height girl: 135.0
Max age boy: 14.0
Max age girl: 15.0
```

17. Να γραφεί πρόγραμμα που θα περιλαμβάνει τις παρακάτω συναρτήσεις:

1. τη συνάρτηση ReadData που θα διαβάζει ακέραιους και θα τους καταχωρεί σε δισδιάστατο πίνακα
2. τη συνάρτηση printTab_2d που θα εμφανίζει τον δισδιάστατο πίνακα.
3. τη συνάρτηση FindMaxRowMinCol που θα δέχεται έναν δισδιάστατο πίνακα, τον αριθμό μιας γραμμής, τον αριθμό μιας στήλης και θα επιστρέφει μέσω κατάλληλων παραμέτρων το μεγαλύτερο στοιχείο της γραμμής και το μικρότερο στοιχείο της στήλης

   Η main θα εκτελεί τα εξής: α) θα καλεί τη συνάρτηση ReadData για πίνακα 3X4, β) θα εμφανίζει τον πίνακα, γ) θα διαβάζει τον αριθμό της γραμμής και της στήλης θα καλεί τη συνάρτηση FindMaxRowMinCol και στη συνέχεια θα εμφανίζει το μεγαλύτερο στοιχείο της γραμμής και το μικρότερο στοιχείο της στήλης.

   Δίνεται ένα στιγμιότυπο εκτέλεσης:

```
Give element 0, 0 10
Give element 0, 1 13
Give element 0, 2 2
Give element 0, 3 17
Give element 1, 0 7
Give element 1, 1 5
Give element 1, 2 12
Give element 1, 3 7
Give element 2, 0 19
Give element 2, 1 14
Give element 2, 2 9
Give element 2, 3 1
  10  13   2  17
   7   5  12   7
  19  14   9   1
Give row 1
Give column 3
max=12, min=1
```

18. Ενας μαθητής θεωρείται ότι απέτυχε αν ο ΜΟ της βαθμολογίας του σε όλα τα μαθήματα είναι μικρότερος του 9.5 και αρίστευσε αν ΜΟ της βαθμολογίας του σε όλα τα μαθήματα είναι μεγαλύτερος από 18.5. Να γραφεί πρόγραμμα για 30 μαθητές και 3 μαθήματα και θα περιλαμβάνει τις παρακάτω συναρτήσεις:

1. τη συνάρτηση populate που θα καταχωρεί σε δισδιάστατο πίνακα τυχαίους αριθμούς από 0 έως και 20. (βαθμολογίες μαθητών). Η συνάρτηση populate διαβάζει την τιμή της παραμέτρου seed για την srand(seed).
2. τη συνάρτηση printTab_2d που θα εμφανίζει τον δισδιάστατο πίνακα
3. τη συνάρτηση FindStudents που θα υπολογίζει και θα επιστρέφει μέσω κατάλληλων παραμέτρων το πλήθος και το ποσοστό των μαθητών που αρίστευσαν, όπως και το πλήθος και το ποσοστό των μαθητών που απέτυχαν.

Δίνεται ένα στιγμιότυπο εκτέλεσης

```
 Give seed:50
  12  19  16
  19   2  17
   1   5   9
  19  20   0
   9  18  10
   8  14  12
  17   9  16
  18   1  18
  11   7   7
   0   3  15
   1  10   3
   4   6  14
  18   3  19
   6   2  12
  11  20   2
   5   2   9
  17  19  11
  14  12   1
   3   0   6
  19  11  18
   0   1  18
   8  14   3
  17  10   2
  10  13   4
  18  17  14
  19   6  19
  13  18   5
   1   1   2
   6  18   2
  10   9  10
Succeed: 0, 0.0%
Fail: 14, 46.7%
```