ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ

Τμήμα Εφαρμοσμένης Πληροφορικής

ΔΙΑΔΙΚΑΣΤΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Εξάμηνο Α'

Φύλλο Ασκήσεων 5 – ΠΙΝΑΚΕΣ

Διδάσκοντες: Μάγια Σατρατζέμη, Αλέξανδρος Χατζηγεωργίου, Στέλιος Ξυνόγαλος, Ηλίας Σακελλαρίου, Αλέξανδρος Καρακασίδης

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. Για την εμφάνιση των τιμών μεταβλητών τύπου `double/float` τα δεδομένα θα εμφανίζονται με ένα δεκαδικό ψηφίο. Σε περίπτωση που απαιτείται διαφορετική στοίχιση ή διαφορετική ακρίβεια θα δίνονται οι απαραίτητες διευκρινήσεις.
5. Για την εμφάνιση πολλών δεδομένων στην ίδια γραμμή θα τυπώνεται ένας κενός χαρακτήρας ανά δεδομένο. Για την αναπαράσταση του κενού χαρακτήρα στις εκφωνήσεις χρησιμοποιείται η κάτω παύλα -underscore- "_".

---

1. Δίνονται για καθένα από `N` άτομα (`N`=γνωστό και `N<=50`) το ποσό των μηνιαίων αποδοχών τους (`long`). Να βρεθεί, για καθένα ξεχωριστά, ο ελάχιστος αριθμός χαρτονομισμάτων (`long`) και κερμάτων (`long`) που χρειάζονται για να συγκεντρωθεί ακριβώς το ποσό και να εμφανιστούν με δεξιά στοίχιση σύμφωνα με την παρακάτω μορφή:

Στήλες:

1-3	4-14	15-21	22-28	29-35	36-42	43-49	50-56	57-63	64-70	71-77	78-84	85-91

A/A	**POSO**	**10000**	**5000**	**1000**	**500**	**200**	**100**	**50**	**20**	**10**	**5**	**1**
1	568	0	0	0	1	0	0	1	0	1	1	3
..	..	..	..	..	..	..	..	..	..	..	..	..
N	..	..	..	..	..	..	..	..	..	..	..	..
SYN	..	..	..	..	..	..	..	..	..	..	..	..

2. Να γίνει πρόγραμμα το οποίο θα υπολογίζει και θα εμφανίζει το ποσό του φόρου (`long`) που πρέπει να πληρώσει ένας φορολογούμενος αν δίνεται το ποσό των καθαρών ετήσιων αποδοχών του (`long`). Ο παρακάτω πίνακας με την φορολογική κλίμακα θεωρείται δοσμένος και σταθερός.

ΚΛΙΜΑΚΙΟ	**%**	**ΦΟΡΟΣ ΚΛΙΜΑΚΟΣ**
1..10000	0	Ποσό Αφορολόγητο
10001..39000	18	5220
39001..49000	21	7320
49001..59000	24	9720
59001..85000	28	17000
85001..100000	33	21950
100001..120000	38	29550
120001..150000	43	42450
150001..170000	49	52250
Υπερβάλλον	50	

Σημείωση: Ο φόρος κλίμακας (τρίτη στήλη) αντιστοιχεί στο μέγιστο ποσό φόρου που μπορεί να πληρώσει κανείς σε μια κατηγορία και χρησιμεύει για τον υπολογισμό της επόμενης κατηγορίας. Αν για παράδειγμα, έχει κανείς εισόδημα 60000€. τότε θα πληρώσει φόρο 9720€ για τα πρώτα 59000€ του εισοδήματός του (ποσό που το παίρνουμε έτοιμο από τη στήλη Φόρος Κλίμακας) και (60000 - 59000) x 28% = 280€, άρα συνολικά 9720€ + 280€ = 10000€. Ένα παράδειγμα εμφάνισης των αποτελεσμάτων, που προκύπτει για εισόδημα 60000€, έχει την παρακάτω μορφή:

```
60000 Euro
9720 Euro
280 Euro
10000 Euro
```

3. Το τρίγωνο του Pascal περιέχει τους συντελεστές του αναπτύγματος του διωνύμου του Νεύτωνα $(x+a)^n$. Να γραφεί πρόγραμμα που θα εμφανίζει τους συντελεστές του αναπτύγματος $(x+a)^n$ όπου `n` γνωστό. Θεωρείστε ότι η μέγιστη τιμή που μπορεί να πάρει το `n` είναι το 11.

Τα αποτελέσματα θα εμφανίζονται σε δεξιά στοίχιση της μορφής:

```
n (2 θέσεις), συντελεστής (5 θέσεις)
```

Ένα παράδειγμα εμφάνισης για `n`=4 έχει την παρακάτω μορφή:

```
0    1
1    1    1
2    1    2    1
3    1    3    3    1
4    1    4    6    4    1
```

4. Μια εταιρεία εμπορεύεται 5 προϊόντα αξίας 25000, 15000, 32000, 21000 και 9200. 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 1	Προϊόν 2	Προϊόν 3	Προϊόν 4	Προϊόν 5
1	10	4	5	6	7
2	7	0	12	1	3
3	4	9	5	0	8
4	3	2	1	5	6

Αν ο πίνακας πωλήσεων είναι *δοσμένος και σταθερός*, να γραφεί πρόγραμμα που θα υπολογίζει:

- το συνολικό ποσό είσπραξης (`long`) για κάθε πωλητή. Η εμφάνιση των ποσών για όλους τους πωλητές θα γίνει στην ίδια γραμμή με ένα κενό χαρακτήρα μεταξύ κάθε ποσού.
- τη συνολική προμήθεια (`double`) για κάθε πωλητή, αν ο κάθε πωλητής έχει προμήθεια 10% επί των εισπράξεων του από πωλήσεις. Η εμφάνιση των προμηθειών για όλους τους πωλητές θα γίνει στην ίδια γραμμή με ένα κενό χαρακτήρα μεταξύ κάθε ποσού.
- τις ποσότητες (`int`) που πουλήθηκαν από κάθε προϊόν. Η εμφάνιση των ποσοτήτων για όλα τα προϊόντα θα γίνει στην ίδια γραμμή με ένα κενό χαρακτήρα μεταξύ κάθε ποσού.

Παράδειγμα εκτέλεσης

```
Synoliko Poso Eispaksis / Pwlhth: 660400 607600 468600 297200
Promitheia / Pwlhth: 66040.00 60760.00 46860.00 29720.00
Posothtes Proiontwn: 24 15 23 12 24
```

5. Να γραφεί πρόγραμμα το οποίο θα διαβάζει έναν αριθμό `M` (`int`) και έναν αριθμό N (`int`) και στη συνέχεια θα διαβάζει τα `MxN` (γραμμές x στήλες) στοιχεία (`long`) ενός διδιάστατου πίνακα Α. Η μέγιστη διάσταση πίνακα Α είναι 100x100, δηλαδή ο χρήστης θα χρησιμοποιεί *μέρος* του πίνακα Α και θεωρείστε ότι ο χρήστης δίνει πάντα Μ και Ν μικρότερα του 100. Στη συνέχεια θα υπολογίζει και θα εμφανίζει τα αθροίσματα των στοιχείων κάθε γραμμής και στήλης. Αν ο πίνακας που εισήγαγε ο χρήστης είναι τετραγωνικός (M=N) τότε θα εμφανίζει και το άθροισμα των δύο διαγώνιων.

Η έξοδος θα εμφανίζει τα στοιχεία του πίνακα και σε κάθε γραμμή το άθροισμα της γραμμής, ενώ σε κάθε στήλη το άθροισμα της στήλης, όπως φαίνεται παρακάτω:

Ενδεικτικά στιγμιότυπα εκτέλεσης προγράμματος

```
Dwse ton arithmo twn grammwn: 2
Dwse ton arithmo twn sthlwn: 3
Thesi [0,0]:10
Thesi [0,1]:20
Thesi [0,2]:30
Thesi [1,0]:1
Thesi [1,1]:2
Thesi [1,2]:3
Table:
  10  20  30| = 60
   1   2   3| = 6
------------
  11  22  33
```

```
Dwse ton arithmo twn grammwn: 3
Dwse ton arithmo twn sthlwn: 3
Thesi [0,0]:1
Thesi [0,1]:2
Thesi [0,2]:3
Thesi [1,0]:10
Thesi [1,1]:20
Thesi [1,2]:30
Thesi [2,0]:100
Thesi [2,1]:200
Thesi [2,2]:300
Table:
   1   2   3 | = 6
  10  20  30 | = 60
 100 200 300 | = 600
------------
 111 222 333
Sum Diag 1: 321,  Diag 2: 123
```

6. Μία μέθοδος για την κατασκευή ενός μαγικού τετραγώνου διαστάσεων 4x4 για 2 δοσμένους ακέραιους αριθμούς `A` και `B` περιγράφεται στον παρακάτω πίνακα:

A	A + 14	B	A + 3
B - 2	A + 5	A + 6	A + 8
A + 7	B - 4	A + 10	A + 4
A + 12	A + 2	A + 1	B + 2

Να γραφεί πρόγραμμα το οποίο θα διαβάζει τις τιμές των μεταβλητών `A` (`long`) και `B` (`long`) και στη συνέχεια θα υπολογίζει και θα εμφανίζει το αντίστοιχο μαγικό τετράγωνο και θα επαληθεύει ότι είναι μαγικό.

Σημείωση: Ένας πίνακας αποτελεί μαγικό τετράγωνο όταν τα αθροίσματα των στοιχείων των στηλών, των γραμμών της πρωτεύουσας και της δευτερεύουσας διαγωνίου είναι ίσα μεταξύ τους.

Τα αποτελέσματα που θα εμφανίζονται είναι:

Τα στοιχεία του μαγικού τετραγώνου κατά γραμμές με ένα κενό χαρακτήρα ανά στήλη, το άθροισμα κάθε γραμμής, κάθε στήλης, των 2 διαγωνίων (όπως φαίνεται στο παρακάτω στιγμιότυπο).

Το μήνυμα 'MAGIC' ή 'NOT MAGIC' ανάλογα με το αν ο πίνακας είναι μαγικός ή όχι.

Ενδεικτικό στιγμιότυπο εκτέλεσης προγράμματος

```
Dwse thn timh tou a:10
Dwse thn timh tou b:20
10 24 20 13 | 67
18 15 16 18 | 67
17 16 20 14 | 67
22 12 11 22 | 67
------------
67 67 67 67
67, 67
MAGIC
```

7. Να γραφεί πρόγραμμα το οποίο θα διαβάζει 4 αριθμούς (`int`) `M1, N1` και `M2, N2`. Στη συνέχεια θα διαβάζει τα `M1xN1` (γραμμέςχστήλες) στοιχεία (`long`) του διδιάστατου πίνακα `A` και θα διαβάζει τα `M2xN2` (γραμμέςχστήλες) στοιχεία (`long`) του διδιάστατου πίνακα `B` (μέγιστες διαστάσεις πινάκων 50x50). Στη συνέχεια να υπολογίζει και να εμφανίζει το άθροισμα `(A+B)`, τη διαφορά `(A-B)` και το γινόμενό `(A*B)` των πινάκων. Να γίνεται έλεγχος αν οι αντίστοιχες πράξεις μπορούν να πραγματοποιηθούν (αν δηλαδή

ταιριάζουν οι διαστάσεις των πινάκων). Σε περίπτωση που δεν ταιριάζουν οι διαστάσεις να εμφανίζεται το μήνυμα 'ERROR' και τα σύμβολα '+', '-', '*' ανάλογα με το ποια πράξη δεν μπορεί να εκτελεστεί.
Το πρόγραμμα θα εμφανίζει τους πίνακες με τα αποτελέσματα των παραπάνω διαδοχικών πράξεων. Τα στοιχεία των πινάκων με τα αποτελέσματα θα εμφανίζονται κατά γραμμές με ένα κενό χαρακτήρα μεταξύ των στοιχείων κάθε στήλης. Συγκεκριμένα, θα εμφανίζονται με την εξής σειρά:

- τα αποτελέσματα της πρόσθεσης ή το μήνυμα 'ERROR +' αν η πράξη δεν είναι εφικτή
- τα αποτελέσματα της αφαίρεσης ή το μήνυμα 'ERROR -' αν η πράξη δεν είναι εφικτή
- τα αποτελέσματα του πολλαπλασιασμού ή το μήνυμα 'ERROR *' αν η πράξη δεν είναι εφικτή

Ενδεικτικό στιγμιότυπο εκτέλεσης προγράμματος

```
Dwse to plhthos twn grammwn m1: 2
Dwse to plhthos twn sthlwn n1: 3
Dwse to plhthos twn grammwn m2: 3
Dwse to plhthos twn sthlwn n2: 3
1
2
3
4
5
6
10
20
30
40
50
60
70
80
90
ERROR +
ERROR -
300 360 420
660 810 960
```

8. Να γίνουν *2 ξεχωριστά προγράμματα* τα οποία θα διαβάζουν 2 αριθμούς (`int`) `M` και `N` στη `main` και καθένα από τα οποία θα επιτελεί τις παρακάτω λειτουργίες:

a) Θα διαβάζει τα `MxN` (γραμμέςχστήλες) στοιχεία (`long`) του διδιάστατου πίνακα `A` (μέγιστη διάσταση πίνακα `A` 50x50) σε μια διαδικασία `Read_Array`. Στη συνέχεια, θα μεταφέρει γραμμή προς γραμμή τα στοιχεία του πίνακα `A` σε ένα μονοδιάστατο πίνακα `B` με `MxN` στοιχεία (μέγιστη διάσταση πίνακα `B` 250) χρησιμοποιώντας μια διαδικασία `Create_Array`. Τέλος, θα εμφανίζει τα στοιχεία του πίνακα `B` σε μια γραμμή με ένα κενό χαρακτήρα μεταξύ τους σε μια διαδικασία `Print_Array`.

b) Θα διαβάζει τα MxN στοιχεία ενός μονοδιάστατου πίνακα B (long) (μέγιστη διάσταση 250) σε μια διαδικασία Read_Array. Στη συνέχεια, θα μεταφέρει τα στοιχεία του πίνακα B στο διδιάστατο πίνακα A (μέγιστη διάσταση 50x50) κατά γραμμές (M γραμμές και N στήλες) χρησιμοποιώντας μια διαδικασία `Create_Array`. Τέλος, θα εμφανίζει τον πίνακα A κατά γραμμές με ένα κενό χαρακτήρα μεταξύ των στοιχείων κάθε γραμμής σε μια διαδικασία `Print_Array`.

Παραδείγματα Εκτέλεσης.
Εκτέλεση προγράμματος (α)

```
Dwse to plhthos twn grammwn: 2
Dwse to plhthos twn sthlwn: 3
1
2
3
4
5
6
1 2 3 4 5 6
```

Εκτέλεση Προγράμματος (β)

```
Dwse thn timh tou m: 2
Dwse thn timh tou n: 3
1
2
3
4
5
6
1 2 3
4 5 6
```

9. Να γραφεί πρόγραμμα το οποίο θα διαβάζει 2 αριθμούς (`int`) `M` και `N` στη `main`, και στη συνέχεια:

- χρησιμοποιώντας μια διαδικασία `Read_Array` θα διαβάζει τα `MxN` (γραμμέςχστήλες) στοιχεία (`long`) ενός διδιάστατου πίνακα `A` (μέγιστη διάσταση 50x50).
- χρησιμοποιώντας μια διαδικασία `Find_Min_of_Rows` θα βρίσκει και θα αποθηκεύει σε ένα μονοδιάστατο πίνακα `B` (μέγιστη διάσταση 50) τον ελάχιστο όρο κάθε γραμμής του πίνακα Α.
- χρησιμοποιώντας μια διαδικασία `Print_Min_Array` θα εμφανίζει τον μονοδιάστατο πίνακα που δημιουργήθηκε στην προηγούμενη διαδικασία.

Οι διαδικασίες θα καλούνται από την `main` με την παραπάνω σειρά.

Ενδεικτικό στιγμιότυπο εκτέλεσης προγράμματος

```
Dwse thn timh tou m: 3
Dwse thn timh tou n: 4
1
2
3
4
40
30
20
10
200
100
300
400
1
10
100
```

10. Μια ναυτιλιακή εταιρεία μεταφέρει οικιακές συσκευές τυποποιημένου μεγέθους και χρησιμοποιεί ειδικά μεταφορικά κιβώτια (containers) τα οποία χωρούν 1, 5, 20 και 50 οικιακές συσκευές. Γράψτε ένα πρόγραμμα το οποίο θα διαβάζει τον αριθμό των οικιακών συσκευών (`long`) που πρόκειται να μεταφερθούν και στη συνέχεια να υπολογίζει και να εμφανίζει τον απαιτούμενο αριθμό μεταφορικών κιβωτίων από κάθε μέγεθος (`long`) έτσι ώστε η μεταφορά να πραγματοποιηθεί με τον πλέον οικονομικό τρόπο (τον ελάχιστο δυνατό αριθμό μεταφορικών κιβωτίων χωρίς αχρησιμοποίητο χώρο). *Η άσκηση θα λυθεί με χρήση πινάκων*.

Η εμφάνιση των αποτελεσμάτων θα γίνει με αύξουσα διάταξη του μεγέθους των containers και ως εξής:
2 θέσεις για την εμφάνιση του τυποποιημένου μεγέθους
5 θέσεις για την εμφάνιση του πλήθους κιβωτίων

Ενδεικτικό στιγμιότυπο εκτέλεσης προγράμματος:

```
Dwse plithos syskeuwn: 1169
 1     4
 5     3
20     0
50    23
```

11. Η Ε.Μ.Υ. καταγράφει ανά 8 ώρες τις θερμοκρασίες 10 πόλεων της Ελλάδος (τα ονόματα των πόλεων είναι κωδικοποιημένα με αριθμούς: 0=Θεσσαλονίκη, 1=Αθήνα κλπ). Να γραφεί πρόγραμμα που θα διαβάζει τις θερμοκρασίες (μεταβλητές τύπου `double`) ενός εικοσιτετραώρου για κάθε πόλη (θερμοκρασία 0ου 8ώρου, 1ου 8ώρου, 2ου 8ώρου για κάθε πόλη). Στη συνέχεια να υπολογίζει και να εμφανίζει:

- τον εθνικό μέσο όρο θερμοκρασίας (μεταβλητή τύπου `double`) και
- για κάθε πόλη τη μέση θερμοκρασία της και τη μέγιστη απόκλισή της από τον εθνικό μέσο όρο (μεταβλητές τύπου `double`).

Τα αποτελέσματα να εμφανίζονται με την παρακάτω μορφή:

εθνικός μέσος όρος θερμοκρασίας
City 0 μέση θερμοκρασία 0ης πόλης_μέγιστη απόκλιση της 0ης πόλης από τον εθνικό μέσο όρο
City 1 μέση θερμοκρασία 1ης πόλης_μέγιστη απόκλιση της 1ης πόλης από τον εθνικό μέσο όρο
.....
City 9 μέση θερμοκρασία 9ης πόλης_μέγιστη απόκλιση της 9ης πόλης από τον εθνικό μέσο όρο

```
Temp of City 0 During 0 8-hour period: 5.6
Temp of City 0 During 1 8-hour period: 5.8
Temp of City 0 During 2 8-hour period: 5.8
Temp of City 1 During 0 8-hour period: 6.9
Temp of City 1 During 1 8-hour period: 6.5
Temp of City 1 During 2 8-hour period: 6.8
Temp of City 2 During 0 8-hour period: 6.4
Temp of City 2 During 1 8-hour period: 6.3
Temp of City 2 During 2 8-hour period: 6.2
Temp of City 3 During 0 8-hour period: 6.5
Temp of City 3 During 1 8-hour period: 3.2
Temp of City 3 During 2 8-hour period: 6.5
Temp of City 4 During 0 8-hour period: 6.3
Temp of City 4 During 1 8-hour period: 6.2
Temp of City 4 During 2 8-hour period: 3.0
Temp of City 5 During 0 8-hour period: 3.6
Temp of City 5 During 1 8-hour period: 3.5
Temp of City 5 During 2 8-hour period: 3.0
Temp of City 6 During 0 8-hour period: 9.6
Temp of City 6 During 1 8-hour period: 6.9
Temp of City 6 During 2 8-hour period: 6.8
Temp of City 7 During 0 8-hour period: 3.2
Temp of City 7 During 1 8-hour period: 3.2
Temp of City 7 During 2 8-hour period: 3.0
Temp of City 8 During 0 8-hour period: 6.0
Temp of City 8 During 1 8-hour period: 6.5
Temp of City 8 During 2 8-hour period: 6.5
Temp of City 9 During 0 8-hour period: 6.4
Temp of City 9 During 1 8-hour period: 6.4
Temp of City 9 During 2 8-hour period: 6.7
5.6
City 0  5.7  0.2
City 1  6.7  1.3
City 2  6.3  0.8
City 3  5.4  2.4
City 4  5.2  2.6
City 5  3.4  2.6
City 6  7.8  4.0
City 7  3.1  2.6
City 8  6.3  0.9
City 9  6.5  1.1
```

12. Η βαθμολόγηση ενός διαγωνίσματος γίνεται με μια συγκεκριμένη μέθοδο. Καθένα από τα `N` θέματα βαθμολογείται στην κλίμακα από 1-100. Στη συνέχεια, ο μέσος βαθμός ανάγεται στην εικοσαβάθμια

κλίμακα. Να κατασκευάσετε ένα πρόγραμμα το οποίο θα διαβάζει το πλήθος των μαθητών `M` (`M` <=30), το πλήθος των θεμάτων `N` (`N`<=10), και τις βαθμολογίες (`double`) των `M` μαθητών για καθένα από τα `N` θέματα. Στη συνέχεια θα υπολογίζει και θα εμφανίζει σε διαφορετική γραμμή για κάθε μαθητή:

- τις βαθμολογίες του και το μέσο όρο βαθμολογίας (`double`) στην 100βάθμια κλίμακα και
- τις βαθμολογίες του (`double`) και το μέσο όρο βαθμολογίας (`double`) στην 20βάθμια κλίμακα

Συγκεκριμένα, για κάθε μαθητή θα εμφανίζονται (με δεξιά στοίχιση 6 χαρακτήρων με 1 δεκαδικό ψηφίο: %6.1lf) οι παραπάνω πληροφορίες σε δύο διαφορετικές γραμμές ως εξής:

βαθμός 1ου θέματος_....._βαθμός Νου θέματος _μέσος όρος (όλα στην 100βάθμια κλίμακα)
βαθμός 1ου θέματος_....._βαθμός Νου θέματος _μέσος όρος (όλα στην 20βάθμια κλίμακα)
Οι βαθμολογίες αποθηκεύονται σε μεταβλητές τύπου double.

```
Dwse ton arithmo twn mathitwn: 3
Dwse ton arithmo twn thematwn: 2
Dwse th bathmologia tou 1ou thematos tou 1ou mathiti: 100
Dwse th bathmologia tou 2ou thematos tou 1ou mathiti: 90
Dwse th bathmologia tou 1ou thematos tou 2ou mathiti: 80
Dwse th bathmologia tou 2ou thematos tou 2ou mathiti: 65
Dwse th bathmologia tou 1ou thematos tou 3ou mathiti: 80
Dwse th bathmologia tou 2ou thematos tou 3ou mathiti: 70
 100.0  90.0  95.0
  20.0  18.0  19.0
  80.0  65.0  72.5
  16.0  13.0  14.5
  80.0  70.0  75.0
  16.0  14.0  15.0
Πιέστε ένα πλήκτρο για συνέχεια. . .
```

13. Να γραφεί πρόγραμμα το οποίο:
(α) Θα ζητά από τον χρήστη δύο θετικούς ακέραιους αριθμούς R (γραμμές) και C (στήλες) από το 1 μέχρι και το 10. Μπορείτε να υποθέσετε με ασφάλεια ότι ο χρήστης θα εισάγει σωστούς αριθμού σύμφωνα με τα προηγούμενα όρια. Δεν απαιτείται έλεγχος. Η εισαγωγή θα γίνεται από την συνάρτηση `main`.
(β) Θα "γεμίζει" τα στοιχεία R x C ενός διδιάστατου πίνακα `A` ακεραίων (διάστασης 10x10) με *τυχαίους ακέραιους* από 0 έως και 99 (`rand() % 100`), μέσω κλήσης μιας συνάρτησης `populate_data`. Δηλαδή το πρόγραμμα θα χρησιμοποιεί *μέρος* του πίνακα Α, βάσει των αριθμών R και C, που εισήγαγε ο χρήστης.
(γ) Θα καλεί μια συνάρτηση `print_array` που θα τυπώνει τα R x C στοιχεία του πίνακα Α.
(δ) Θα καλεί μια διαδικασία `change_array`, η οποία **για κάθε γραμμή** του πίνακα Α θα θέτει όσα στοιχεία είναι *αριστερά* του *μέγιστου* της γραμμής ίσα με αυτό (δηλαδή όσα είναι στην ίδια γραμμή με αυτό, αλλά έχουν μικρότερο δείκτη στήλης θα γίνουν ίσα με το μέγιστο).
(ε) Τέλος, θα εμφανίζονται και πάλι τα στοιχεία του "αλλαγμένου" πίνακα κατά γραμμές.
Όλες οι συναρτήσεις σας θα ακολουθούν το πρότυπο C99 για πίνακες 2 διαστάσεων. ΠΡΟΣΟΧΗ στις δηλώσεις των παραμέτρων των συναρτήσεων. Είναι άλλο η διάσταση του πίνακα, και άλλο ποιο μέρος του πίνακα έχει χρησιμοποιηθεί. θα πρέπει να περάσετε περισσότερες παραμέτρους.

Ενδεικτικό στιγμιότυπο εκτέλεσης προγράμματος

```
Dwse ton arithmo twn grammwn: 3
Dwse ton arithmo twn sthlwn: 4
ARXIKOS PINAKAS
41 67 34  0
69 24 78 58
62 64  5 45
ALLAGMENOS PINAKAS
67 67 34  0
78 78 78 58
64 64  5 45
```

14. Να γραφεί πρόγραμμα που θα διαβάζει το πλήθος των στοιχείων `N` και τα στοιχεία (`long`) ενός μονοδιάστατου πίνακα (μέγιστη διάσταση 50) και θα εμφανίζει τον πίνακα. Στη συνέχεια θα υπολογίζει και θα εμφανίζει το μέγιστο στοιχείο του πίνακα, πόσες φορές εμφανίζεται και τις θέσεις στις οποίες αυτό εμφανίζεται. Θα λύσετε το πρόβλημα με μία μόνο "σάρωση" του πίνακα. Η λύση απαιτεί την υλοποίηση των εξής συναρτήσεων:

a. Συνάρτηση που θα διαβάζει και θα επιστρέφει έναν ακέραιο αριθμό.
b. Συνάρτηση που θα διαβάζει τα στοιχεία ενός μονοδιάστατου πίνακα (χρήση πρότυπου C99)
c. Συνάρτηση που θα εμφανίζει τα στοιχεία μονοδιάστατου πίνακα (όλα τα στοιχεία του πίνακα στην ίδια γραμμή με ένα κενό μεταξύ τους). (χρήση πρότυπου C99)
d. Συνάρτηση που θα υπολογίζει το μέγιστο στοιχείο ενός μονοδιάστατου πίνακα, θα εμφανίζει το μέγιστο και θα επιστρέφει πόσες φορές εμφανίζεται το μέγιστο και τις θέσεις στις οποίες αυτό εμφανίζεται. Δίνεται το πρότυπο της συνάρτησης αυτής:
`int FindMaxsPos(int size, long a[], int index[])`

Ενδεικτικό στιγμιότυπο εκτέλεσης προγράμματος

```
Dwse to plhthos twn stoixeiwn: 10
5
8
9
2
3
5
15
9
15
4
5 8 9 2 3 5 15 9 15 4
maximum 15
Plithos maximum 2
Positions of maximum 6 8
```

15. Το παρακάτω πρόγραμμα εκτελεί τα εξής: Γεμίζει με τυχαίους αριθμούς έναν πίνακα a 100 θέσεων μνήμης. Στη συνέχεια εμφανίζει τα στοιχεία του πίνακα a (όλα τα στοιχεία του a στην ίδια γραμμή με ένα κενό μεταξύ τους). Διαβάζει έναν ακέραιο αριθμό ar και ελέγχει και καταχωρεί σε ένα νέο πίνακα b όλους τους αριθμούς του πίνακα a που είναι μεγαλύτεροι από τον αριθμό ar. Εμφανίζει τα στοιχεία του πίνακα b (όλα τα στοιχεία του b στην ίδια γραμμή με ένα κενό μεταξύ τους).

Να γραφεί νέα έκδοση του παρακάτω προγράμματος με χρήση συναρτήσεων.

a) Συνάρτηση που θα γεμίζει με τυχαίους αριθμούς έναν πίνακα a 100 θέσεων μνήμης.
b) Συνάρτηση που θα εμφανίζει τα στοιχεία μονοδιάστατου πίνακα (όλα τα στοιχεία του πίνακα στην ίδια γραμμή με ένα κενό μεταξύ τους).
c) Συνάρτηση που θα διαβάζει έναν ακέραιο αριθμό.
d) Συνάρτηση που θα ελέγχει και θα καταχωρεί σε ένα νέο πίνακα b όλους τους αριθμούς του πίνακα a που είναι μεγαλύτεροι από τον αριθμό ar.

Η συνάρτηση main θα καλεί κατάλληλα τις παραπάνω συναρτήσεις ώστε να εκτελεί το ίδιο έργο με την έκδοση 1 του παρακάτω προγράμματος.

```
#include <stdio.h>
#include <stdlib.h>
#include "simpio.h"

#define elements 100

main()
{
   int a[elements],b[elements];
   int i,ar,k;

   for (i=0;i<elements;i++)
      a[i]=rand();

    printf("\n");
    for (i=0;i<elements;i++)
        printf("%d ",a[i]);
```

```
        printf("\nDose arithmo:");
        ar=GetInteger();

        k=0;
        for (i=0;i<elements;i++)
        {
           if (a[i]>ar)
           {
                b[k]=a[i];
                k++;
           }
        }

        printf("\nplithos=%d\n",k);

        for (i=0;i<k;i++)
        {
            printf("%d ",b[i]);
        }
        printf("\n");
       system("pause");
    }
```

16. Να αναπτυχθεί (με τη χρήση των συναρτήσεων που περιγράφονται) πρόγραμμα στο οποίο,
- θα γεμίζει τον πίνακα a μεγέθους 50 με τυχαίους αριθμούς από το 0 έως το 9 χρησιμοποιώντας την συνάρτηση `rand()` (`a[i]=rand() % 10`), της βιβλιοθήκης `<stdlib.h>` (συνάρτηση `populate`)
- θα εμφανίζει τον πίνακα a στην οθόνη (συνάρτηση `printArr`),
- θα ζητά έναν ακέραιο αριθμό N από το 0 έως και το 9 από τον χρήστη - δεν απαιτείται έλεγχος,
- θα καλεί την συνάρτηση `checkTable`, η οποία θα **υπολογίζει** τον αριθμό των εμφανίσεων του αριθμού N στον πίνακα a και τις θέσεις του πίνακα στις οποίες εμφανίζεται ο αριθμός
- θα εμφανίζει τα αποτελέσματα στο χρήστη, όπως φαίνεται στο παράδειγμα εκτέλεσης. Η εμφάνιση των αποτελεσμάτων θα γίνεται από τη συνάρτηση `main()` (κυρίως πρόγραμμα), και των θέσεων που εμφανίζεται ο αριθμός N καλώντας την `printArr`.

```
 1  7  4  0  9  4  8  8  2  4  5  5  1  7  1  1  5  2  7  6  1  4  2  3  2  2  1
  6  8  5  7  6  1  8  9  2  7  9  5  4  3  1  2  3  3  4  1  1  3  8
--------------
Dose arithmo apo to 0 ews to 9: 1

o Arithmos 1 emfanizetai 10 fores.
Stis Theseis:
 0  12  14  15  20  26  32  41  46  47
-----------------
Press any key to continue...
```

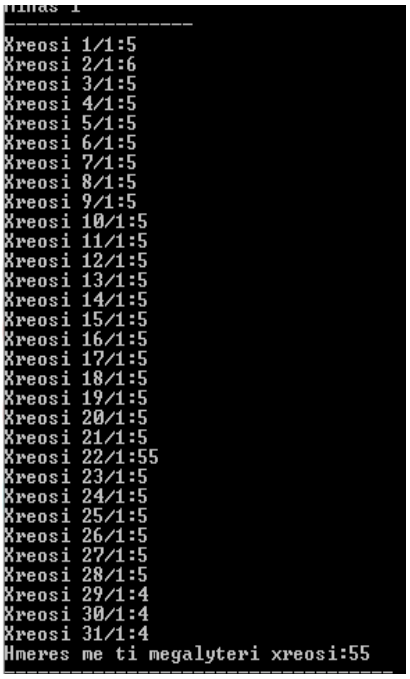
17. Η βαθμολογία σε ένα μάθημα ενός μαθητή δίνεται σε ακέραιες τιμές από το 0 μέχρι και το 20. Ένας μαθητής θεωρείται ότι απέτυχε όταν ο μέσος όρος της βαθμολογίας του σε όλα τα μαθήματα είναι μικρότερος του 9.5, ενώ θεωρείται ότι αρίστευσε αν ο μέσος όρος του είναι μεγαλύτερος ή ίσος του 18.5. Να γραφεί πρόγραμμα το οποίο να διαβάζει τις βαθμολογίες 10 μαθητών σε 3 μαθήματα και να τις καταχωρεί σε κατάλληλο πίνακα. Το πρόγραμμα θα υπολογίζει και να εμφανίζει το πλήθος και το ποσοστό των μαθητών που απέτυχαν, καθώς και το πλήθος και το ποσοστό των μαθητών που αρίστευσαν.

Ενδεικτικό στιγμιότυπο εκτέλεσης προγράμματος

```
Mathitis 1
--------------
```

```
Mathima 1:8
Mathima 2:7
Mathima 3:5
Mathitis 2
--------------
Mathima 1:6
Mathima 2:8
Mathima 3:9
Mathitis 3
--------------
Mathima 1:6
Mathima 2:9
Mathima 3:10
Mathitis 4
--------------
Mathima 1:10
Mathima 2:11
Mathima 3:12
Mathitis 5
--------------
Mathima 1:12
Mathima 2:12
Mathima 3:13
Mathitis 6
--------------
Mathima 1:13
Mathima 2:13
Mathima 3:14
Mathitis 7
--------------
Mathima 1:18
Mathima 2:19
Mathima 3:18
Mathitis 8
--------------
Mathima 1:18
Mathima 2:20
Mathima 3:19
Mathitis 9
--------------
Mathima 1:17
Mathima 2:18
Mathima 3:17
Mathitis 10
--------------
Mathima 1:17
Mathima 2:16
Mathima 3:16
Plithos apotyxonton: 3  30.00%
Plithos aristoyxon: 1  10.00%
```

18. Μια εταιρεία κινητής τηλεφωνίας καταγράφει τις συνολικές χρεώσεις των πελατών της σε καθημερινή βάση για όλο το έτος. Να γραφεί πρόγραμμα το οποίο να διαβάζει τις συνολικές (int) χρεώσεις κάθε ημέρας και να τις καταχωρίζει σε κατάλληλο (int) πίνακα. Το πρόγραμμα να εμφανίζει την ημέρα ή τις ημέρες (ημέρα και μήνα) με τη μεγαλύτερη χρέωση(int max).



(Το παράδειγμα της εικόνας είναι με μικρότερο δείγμα)

19. Να γραφεί ένα πρόγραμμα στο οποίο να δημιουργηθεί ένας πίνακας ο οποίος θα έχει 3 γραμμές και 4 στήλες. Το στοιχείο σε κάθε θέση προκύπτει ως άθροισμα του δείκτη θέσης (index) της γραμμής και της στήλης. Στο τέλος να τυπώνονται τα στοιχεία του πίνακα όπως φαίνεται παρακάτω:

```
0 1 2 3
1 2 3 4
2 3 4 5
```

20. Να γραφεί ένα πρόγραμμα το οποίο να διαβάζει 5 ακεραίους και να τους αποθηκεύει σε έναν πίνακα. Στη συνέχεια, το πρόγραμμα να *περιστρέφει* τα στοιχεία του πίνακα *μία θέση δεξιά* και θα τυπώνει τον πίνακα στην οθόνη. Για παράδειγμα, αν τα στοιχεία του πίνακα είναι: 1, -9, 5, 3, 4 ο πίνακας μετά την περιστροφή θα είναι: 4, 1, -9, 5, 3. Η εκτέλεση φαίνεται παρακάτω.
Παράδειγμα εκτέλεσης:

```
Enter number: 1
Enter number: -9
Enter number: 5
Enter number: 3
Enter number: 4
4 1 -9 5 3
```

21. Να γραφεί ένα πρόγραμμα το οποίο να διαβάζει 5 πραγματικούς αριθμούς τύπου `double` και να τους αποθηκεύει σε έναν πίνακα. Το πρόγραμμα να υπολογίζει την απόσταση μεταξύ διαδοχικών στοιχείων του πίνακα και την αποθηκεύει σε ένα νέο πίνακα (θα καθορίσετε εσείς το απαιτούμενο μέγεθος). Η απόσταση προκύπτει ως η απόλυτη τιμή της διαφοράς των στοιχείων (συνάρτηση `fabs`). Για παράδειγμα, αν τα τέσσερα πρώτα στοιχεία του πίνακα είναι: 5.2, -3.2, 7.5, 12.22, οι αντίστοιχες αποστάσεις είναι: |-3.2-5.2| = 8.4, |7.5-(-3.2)| = 10.7 και |12.22-7.5| = 4.72. Το πρόγραμμα θα εμφανίζει στο τέλος τον αρχικό πίνακα που εισήγαγε ο χρήστης, σε μια γραμμή καθώς και τον πίνακα διαφορών σύμφωνα με την εκτέλεση που παρουσιάζεται. Κάθε στοιχείο των πινάκων θα εμφανίζεται με ακρίβεια 2 δεκαδικών ψηφίων.

Παράδειγμα Εκτέλεσης

```
Enter number (0):5.2
Enter number (1):-3.2
Enter number (2):7.5
Enter number (3):12.22
Enter number (4):12.22
5.20 -3.20 7.50 12.22 12.22
8.40 10.70 4.72 0.00
```

22. Γράψτε ένα πρόγραμμα που θα ελέγχει την εγκυρότητα των αριθμών πιστωτικών καρτών. Το πρόγραμμα θα διαβάζει ένα δεκαεξαψήφιο ακεραίο (τύπου `long long` - δείτε διευκρίνιση στο τέλος της άσκησης*) από τον χρήστη (με την συνάρτηση `GetLongLong()`)* και θα τυπώνει τον αριθμό της κάρτας ακολουθούμενο από τη λέξη "VALID" αν αυτός είναι έγκυρος αριθμός πιστωτικής κάρτας ή τη λέξη "invalid" διαφορετικά, σύμφωνα με τα μηνύματα που εμφανίζονται παρακάτω.
Για την υλοποίηση της άσκησης θα δημιουργήσετε ένα πίνακα τύπου `long long`, μεγέθους 16, στον οποίο σε κάθε θέση του πίνακα θα βάλετε ένα ψηφίο του κωδικού.
Ο έλεγχος εγκυρότητας του κάθε δεκαεξαψήφιου αριθμού γίνεται σύμφωνα με τα ακόλουθα:

- Αν το πρώτο ψηφίο είναι μικρότερο από 4 ή μεγαλύτερο από 7, η κάρτα χαρακτηρίζεται **ως μη έγκυρη.**
- Αν ο αριθμός δεν έχει ακριβώς 16 ψηφία τότε η κάρτα είναι **μη έγκυρη**.
- Διαφορετικά:
  - Διπλασιάζουμε (μόνο) τα ψηφία που βρίσκονται στις περιττές θέσεις του αριθμού (δηλαδή τα 1ο, 3ο, 5ο,..., 15ο, όπως φαίνονται στο παράδειγμα εκτέλεσης με κίτρινο χρώμα). Όσα από αυτά γίνουν διψήφια, προσθέτουμε τα ψηφία τους για να γίνουν μονοψήφια. Παραδείγματα:
    - αρχικό ψηφίο **4**, 2*4=8 άρα νέο ψηφίο **8**.
    - αρχικό ψηφίο **7**, 2*7=14 άρα νέο ψηφίο 1+4 =**5**.
  - Υπολογίσουμε το άθροισμα όλων των νέων ψηφίων που προκύπτουν (*δηλαδή τόσο εκείνα σε περιττές θέσεις μετά τις αλλαγές όσο και εκείνα στις άρτιες*). Αν το άθροισμα των 16 ψηφίων είναι ακέραιο πολλαπλάσιο του 10, η κάρτα είναι **έγκυρη**. και σε διαφορετική περίπτωση **μη έγκυρη**.

Παράδειγμα ελέγχου εγκυρότητας:
Αριθμός κάρτας: 4417123456789113

																Άθροισμα
4	**4**	**1**	**7**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**1**	**1**	**3**	
8	**4**	**2**	**7**	**2**	**2**	**6**	**4**	**1**	**6**	**5**	**8**	**9**	**1**	**2**	**3**	**70**

Αριθμός κάρτας: 4012888888881881

																Άθροισμα
4	**0**	**1**	**2**	**8**	**8**	**8**	**8**	**8**	**8**	**8**	**8**	**1**	**8**	**8**	**1**	
8	**0**	**2**	**2**	**7**	**8**	**7**	**8**	**7**	**8**	**7**	**8**	**2**	**8**	**7**	**1**	**90**

Παραδείγματα Εκτέλεσης
Εκτέλεση 1

```
Insert card number:4417123456789113
4417123456789113 is VALID.
```

Εκτέλεση 2

```
Insert card number:4012888888881881
4012888888881881 is VALID.
```

Εκτέλεση 3

```
Insert card number:5610591081018250
5610591081018250 is VALID.
```

Εκτέλεση 4

```
Insert card number:2545698574589652
2545698574589652 is invalid.
```

Εκτέλεση 5 (πολλά ψηφία - 17):

```
Insert card number:54012888888881881
54012888888881881 is invalid.
```

Εκτέλεση 6:

```
Insert card number:1456
1456 is invalid.
```

(*μπορείτε να δοκιμάσετε και την δική σας πιστωτική κάρτα*)

* Δυστυχώς ο compiler των windows χρησιμοποιεί μόνο 4 bytes για την αποθήκευση των `long` (αντίθετα με εκείνον στο linux που χρησιμοποιεί 8 bytes). Κατά συνέπεια οι 16-ψηφιοι αριθμοί είναι εκτός εύρους. Η λύση σε αυτό είναι να χρησιμοποιηθεί ο τύπος `long long`.
  Για την εμφάνιση ενός αριθμού `long long` χρησιμοποιούμε την μορφοποίηση `%lld`, για παράδειγμα

```
long long temp;
temp = 12;
printf(" number is %lld",temp);
```

  Για την ανάγνωση ενός `long long` θα χρησιμοποιήσετε την ακόλουθη συνάρτηση `GetLongLong()` *την οποία θα συμπεριλάβετε στο αρχείο του κώδικά σας ως συνάρτηση του προγράμματός σας* (προφανώς απαιτείται και forward declaration):

```
long long GetLongLong(void)
{
    string line;
    long long value;
    char termch;

    while (TRUE) {
        line = GetLine();
        switch (sscanf(line, " %lld %c", &value, &termch)) {
          case 1:
            FreeBlock(line);
            return (value);
          case 2:
            printf("Unexpected character: '%c'\n", termch);
            break;
          default:
            printf("Please enter an integer\n");
            break;
        }
        FreeBlock(line);
        printf("Retry: ");
    }
}
```

  Ο κώδικας της συνάρτησης είναι στο αρχείο **get_longlong.c** που βρίσκεται στον φάκελο Έγγραφα->2_Εργασίες στο Eclass, ώστε να τον συμπεριλάβετε στο πρόγραμμά σας.

23. Ένα "ενδιαφέρον" τετράγωνο είναι ένας πίνακας ακεραίων 4x4, όπου το άθροισμα των δύο διαγωνίων του, καθώς και το άθροισμα τυχαίων στοιχείων που βρίσκονται *σε διαφορετική γραμμή και στήλη* είναι S. Αν δοθούν δύο σύνολα 4 ακεραίων (8 ακέραιοι συνολικά), το S ισούται με το άθροισμα των 8 αυτών ακεραίων. Η κατασκευή του πίνακα δίνεται παρακάτω. Έστω το σύνολο A με αριθμούς $\alpha_0,\alpha_1,\alpha_2,\alpha_3$ και το σύνολο B με αριθμούς $\beta_0,\beta_1,\beta_2,\beta_3$. Το ενδιαφέρον τετράγωνο με $S = \alpha_0 + \alpha_1 + \alpha_2 + \alpha_3 + \beta_0 + \beta_1 + \beta_2 + \beta_3$, είναι

$\alpha_0+\beta_0$	$\alpha_1+\beta_0$	$\alpha_2+\beta_0$	$\alpha_3+\beta_0$
$\alpha_0+\beta_1$	$\alpha_1+\beta_1$	$\alpha_2+\beta_1$	$\alpha_3+\beta_1$
$\alpha_0+\beta_2$	$\alpha_1+\beta_2$	$\alpha_2+\beta_2$	$\alpha_3+\beta_2$
$\alpha_0+\beta_3$	$\alpha_1+\beta_3$	$\alpha_2+\beta_3$	$\alpha_3+\beta_3$

Να αναπτύξετε ένα πρόγραμμα το οποίο (α) δέχεται τα δύο σύνολα ακεραίων και τα αποθηκεύει σε δύο αντίστοιχους μονοδιάστατους πίνακες ακεραίων με μέγεθος 4 (β) κατασκευάζει το ενδιαφέρον τετράγωνο σε ένα πίνακα ακεραίων 4x4, και εμφανίζει το άθροισμα των 8 ακεραίων και τον πίνακα στην οθόνη. Για παράδειγμα:

```
Insert SetA num 0:1
Insert SetA num 1:3
Insert SetA num 2:5
Insert SetA num 3:2
Insert SetB num 0:7
Insert SetB num 1:6
Insert SetB num 2:9
Insert SetB num 3:10
Sum: 43
     8    10    12     9
     7     9    11     8
    10    12    14    11
    11    13    15    12
```

24. Καλούμε **πολυώνυμο του x** κάθε παράσταση της μορφής:

$$\alpha_v x^v + \alpha_{v-1}x^{v-1} + \ldots + \alpha_1 x + \alpha_0,$$

όπου ν είναι ένας φυσικός αριθμός και $\alpha_0, \alpha_1, \ldots, \alpha_v$ είναι πραγματικοί αριθμοί. Τα μονώνυμα $\alpha_v x^v$, $\alpha_{v-1}x^{v-1}$, …, $\alpha_1 x$, α_0 λέγονται **όροι (terms)** του πολυωνύμου και οι αριθμοί $\alpha_v, \alpha_{v-1}, \ldots, \alpha_1, \alpha_0$ **συντελεστές (coefficients)** αυτού. Ειδικότερα ο α_0 λέγεται **σταθερός όρος** του πολυωνύμου. **Βαθμός (degree)** του πολυωνύμου ονομάζεται η μεγαλύτερη δύναμη του x με μη μηδενικό συντελεστή.

Στην άσκηση που ακολουθεί, για λόγους απλότητας θεωρούμε ότι τα πολυώνυμα είναι:

- **διατεταγμένα** κατά τις φθίνουσες δυνάμεις του x,
- **πλήρη**, δηλαδή δεν τους «λείπουν» όροι, ανάμεσα στο μεγιστοβάθμιο και τον σταθερό όρο, ακόμα και αν έχουν συντελεστή 0
- **έχουν τον ίδιο βαθμό,** ο οποίος δεν ξεπερνά το 9.

Το παρακάτω πρόγραμμα σε C λαμβάνει από το χρήστη τους συντελεστές δύο πολυωνύμων, εκτυπώνει τα δύο πολυώνυμα, υπολογίζει το άθροισμά τους, εκτυπώνει το άθροισμα, και τέλος εκτυπώνει την αριθμητική τιμή κάθε πολυωνύμου καθώς και του πολυωνύμου που προκύπτει από την πρόσθεση των πολυωνύμων, για τιμή του x που δίνεται από το χρήστη.
Ζητείται να ξαναγράψετε τον πηγαίο κώδικα του προγράμματος αξιοποιώντας συναρτήσεις τόσο για τα τμήματα κώδικα που επαναλαμβάνονται όσο και για τα τμήματα κώδικα που υλοποιούν διακριτές λειτουργίες, διατηρώντας τη λειτουργικότητα του αρχικού προγράμματος αναλλοίωτη. *Μπορείτε να συμβουλευτείτε το exec_fun_arrays_Slides.pdf στο φάκελο Lecture6 (αντίστοιχα αρχεία κώδικα exec_fun_arrays_a.c και exec_fun_arrays_b.c.*
Οι εντολές με έντονη γραφή δεν πρέπει να συμπεριληφθούν σε συναρτήσεις και πρέπει να παραμείνουν στη main.
Υπόδειξη1*: απαιτείται η δήλωση και ο ορισμός **4 συνολικά συναρτήσεων**.*
Υπόδειξη2*: Η συνάρτηση double pow(double x, double y) της βιβλιοθήκης <math.h> επιστρέφει το x υψωμένο στη δύναμη y, δηλαδή x^y.*

```
#include <stdio.h>
#include "genlib.h"
```

```
#include "simpio.h"
#include <math.h>

int main() {

    int degree, i;
    int coefficientsA[10], coefficientsB[10], coefficientsSum[10];
    double x, value1, value2, valueSum;

    //Ανάγνωση βαθμού πολυωνύμων
    printf("Ti vathmo exoyn ta polywnyma?\n");
    degree = GetInteger();

    //Ανάγνωση συντελεστών πρώτου πολυωνύμου
    for(i=0; i<=degree; i++) {
        printf("Eisagete to syntelesti tou orou %d: ", i);
        coefficientsA[i] = GetInteger();
    }

    //Εκτύπωση πρώτου πολυωνύμου
    i=degree;
    while(i>=1) {
        printf("%dx^%d + ", coefficientsA[i], i);
        i--;
    }
    printf("%d\n", coefficientsA[0]);

    //Ανάγνωση συντελεστών δεύτερου πολυωνύμου
    for(i=0; i<=degree; i++) {
        printf("Eisagete to syntelesti tou orou %d: ", i);
        coefficientsB[i] = GetInteger();
    }

    //Εκτύπωση δεύτερου πολυωνύμου
    i=degree;
    while(i>=1) {
        printf("%dx^%d + ", coefficientsB[i], i);
        i--;
    }
    printf("%d\n", coefficientsB[0]);

    //πρόσθεση πολυωνύμων
    for(i=0; i<=degree; i++) {
        coefficientsSum[i] = coefficientsA[i] + coefficientsB[i];
    }

    //Εκτύπωση αθροίσματος
    i=degree;
    while(i>=1) {
        printf("%dx^%d + ", coefficientsSum[i], i);
        i--;
    }
    printf("%d\n", coefficientsSum[0]);

    printf("Dwse mia timi gia to x: ");
    x = GetReal();
```

```
    //Εκτύπωση τιμών πολυωνύμων
    value1 = 0;
    for(i=0; i<=degree; i++)
        value1 += coefficientsA[i]*pow(x, (double)i);

    value2 = 0;
    for(i=0; i<=degree; i++)
        value2 += coefficientsB[i]*pow(x, (double)i);

    valueSum = 0;
    for(i=0; i<=degree; i++)
        valueSum += coefficientsSum[i]*pow(x, (double)i);

    printf("Value1: %g  Value2: %g  Value of Sum: %g", value1,value2,valueSum);
return 0;
}
```

25. Οι θέσεις ενός μικρού αεροπλάνου αναπαριστώνται από έναν πίνακα δύο διαστάσεων (`seats`) όπως φαίνεται στο ημιτελές πρόγραμμα C που ακολουθεί (για λόγους απλότητας ο πίνακας είναι αρχικοποιημένος και σας *δίνεται* και η συνάρτηση **upDateSeats** ώστε να δοκιμάσετε και άλλες τιμές, την οποία και θα διατηρήσετε στον τελικό κώδικα).

```
#define ROWS 12
#define COLUMNS 4

void upDateSeats(int row, int col, int tab[row][col]);

int main() {
   int seats[ROWS][COLUMNS] = {{1,0,   0, 1},
                               {1,0,   0, 1},
                               {0,1,   0, 0},
                               {0,0,   1, 1},
                               {1,1,   0, 0},
                               {1,1,   0, 1},
                               {0,0,   1, 1},
                               {1,0,   0, 0},
                               {0,0,   1, 0},
                               {0,0,   0, 0},
                               {0,0,   0, 0},
                               {1,1,   0, 1}};
   upDateSeats(ROWS,COLUMNS,seats);
   return 0;
}

/* Function for Testing. */
void upDateSeats(int rows, int cols, int tab[rows][cols]){
   int i,j;
   printf("Update Seats? (0/no):");
   if(!GetInteger()) return;
   for(i=0;i<rows;i++)
      for(j=0;j<cols;j++)
         tab[i][j] = (printf("seat (%2d,%2d)",i,j),GetInteger());
    printf("\n");
}
```

Αν το περιεχόμενο μιας θέσης του πίνακα έχει την τιμή 0, η αντίστοιχη θέση του αεροπλάνου είναι διαθέσιμη. Αν το περιεχόμενο μιας θέσης του πίνακα έχει την τιμή 1, η αντίστοιχη θέση του αεροπλάνου είναι κατειλημμένη.

Να γραφεί συνάρτηση με όνομα **findEmptyRows** που να εκτυπώνει τους αριθμούς των σειρών που είναι τελείως κενές. *Σημειώνεται ότι η αρίθμηση των σειρών/στηλών του αεροπλάνου ξεκινά από το 0 (ίδια με εκείνη των πινάκων στην C).*
Να γραφεί συνάρτηση με όνομα **findFirstAvailableWindow** που να εκτυπώνει την πρώτη διαθέσιμη θέση (σειρά, στήλη) σε παράθυρο.
Να γραφεί πρόγραμμα που θα καλεί τις παραπάνω συναρτήσεις μέσα στη συνάρτηση **main**.

Το αποτέλεσμα της εκτέλεσης του προγράμματος για τον παραπάνω πίνακα είναι:

```
Update Seats? (0/no):0
Row: 9
Row: 10
First available window seat: 2,0
```