Τμήμα Εφαρμοσμένης Πληροφορικής

ΔΙΑΔΙΚΑΣΤΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Εξάμηνο Α'

Φύλλο Ασκήσεων 6 – ΔΕΙΚΤΕΣ

Διδάσκοντες: Μάγια Σατρατζέμη, Αλέξανδρος Χατζηγεωργίου, Στέλιος Ξυνόγαλος, Θεόδωρος Κασκάλης, 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. *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 το 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 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 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 της άσκησης) δύο τιμές.

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

b. Θα διαβάζει 2 ακέραιους αριθμούς Α, Β όπου Α<Β

c. Θα βρίσκει: (1) το πλήθος των αριθμών (που διάβασε στο ερώτημα a) που είναι μικρότεροι ή ίσοι του Α, (2) το πλήθος των αριθμών που είναι μεγαλύτεροι ή ίσοι του Β, και (3) το πλήθος των αριθμών που είναι μεγαλύτεροι του Α και μικρότεροι του Β.

d. Θα εμφανίζει τους Ν ακέραιους, τους αριθμούς Α και Β, και τις τιμές που προσδιόρισε στο ερώτημα c.

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 5 πόλεις της Ελλάδας. Οι μηνιαίες εισπράξεις κάθε αίθουσας για ένα καλοκαίρι καταχωρούνται σε πίνακα δύο διαστάσεων (5Χ3). Να γραφεί ένα πρόγραμμα που θα εκτελεί τις παρακάτω λειτουργίες με τη χρήση διαδικασιών ή συναρτήσεων

- **a.** Θα διαβάζει τις μηνιαίες εισπράξεις κάθε πόλης για ένα καλοκαίρι και θα τις αποθηκεύει στον πίνακα 5Χ3.
- **b.** Θα υπολογίζει τη μέση μηνιαία είσπραξη για κάθε αίθουσα
- **c.** Θα υπολογίζει τη μέση μηνιαία είσπραξη για κάθε μήνα
- **d.** Θα υπολογίζει τη μικρότερη μηνιαία είσπραξη και σε ποια πόλη και σε ποιο μήνα πραγματοποιήθηκε
- **e.** Θα εμφανίζει τη μέση μηνιαία είσπραξη για κάθε αίθουσα, τη μέση μηνιαία είσπραξη για κάθε μήνα, μικρότερη μηνιαία είσπραξη και σε ποια πόλη και σε ποιο μήνα πραγματοποιήθηκε

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.

Για να λυθεί η άσκηση πρέπει να υπολογίσουμε τη συχνότητα εμφάνισης του κάθε βαθμού.

Το πρόγραμμα θα εκτελεί τις παρακάτω λειτουργίες με τη χρήση διαδικασιών ή συναρτήσεων

- **a.** Θα διαβάζει τις βαθμολογίες των 30 μαθητών και θα τις αποθηκεύει σε πίνακα

b. Θα υπολογίζει τη συχνότητα εμφάνισης του κάθε βαθμού
c. Θα υπολογίζει ποιος βαθμός παρατηρήθηκε τις περισσότερες φορές (μεγαλύτερη συχνότητα και ποιος βαθμός τη παρουσίασε)
d. Θα εμφανίζει τις συχνότητες των βαθμών, το βαθμό που παρατηρήθηκε τις περισσότερες φορές και πόσες φορές παρατηρήθηκε

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
Press Return key to continue: _
```

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>
#include "genlib.h"
#include "simpio.h"

#define ATHLITES 8

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);
        vathmologiaKolymbi[i] = GetReal();
    }

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

    printf("EISAGWGI VATHMOLOGIAS ATHLIMATOS 3 - Treximo\n");
    for(i=0; i<ATHLITES; i++) {
```

```
        printf("Eisagete ti vathmologia tou athliti %d: ", i);
        vathmologiaTreximo[i] = GetReal();
    }

    //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);

    system("PAUSE");
}
```

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 παρατήρηση 5 στην αρχή της άσκησης) 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 ψωφίων. Παραδείγματα εκτέλεσης για διαφορετικές περιπτώσεις αριθμών ακολουθούν παρακάτω:

Παράδειγμα εκτέλεσης 1:

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

Παράδειγμα εκτέλεσης 2:

```
Please insert non-negative number:1453
```

```
Digits: 4
Average: 3.250
Max: 5
```

Παράδειγμα εκτέλεσης 3:

```
Please insert non-negative number:1
Digits: 1
Average: 1.000
Max: 1
```

Παράδειγμα εκτέλεσης 4:

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