ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΛΟΠΟΝΝΗΣΟΥ
Τμήμα Πληροφορικής & Τηλεπικοινωνιών

Ασκήσεις για Επίλυση (Προγραμματισμός Ι) - bonus

Παράδοση: Upload μέσω του e-class (οι ασκήσεις δεν είναι υποχρεωτικές).

Καταληκτική Ημερομηνία/Ώρα Αποστολής: Κυριακή, 14/01/2024 (23:59:59).

Οδηγίες Υποβολής: Ο κώδικας με τη λύση της κάθε άσκησης θα πρέπει να περιέχεται σε ξεχωριστό αρχείο. Το κάθε αρχείο θα πρέπει να ακολουθεί την παρακάτω ονοματολογία: `Epitheto_Onoma_Askisi_X.c`, όπου `Epitheto` και `Onoma` είναι το επίθετο και το όνομα του κάθε φοιτητή με λατινικούς χαρακτήρες και `X` η αντίστοιχη αρίθμηση της άσκησης.

Π.χ. για τον υποτιθέμενο φοιτητή Νίκο Κούλη, η απάντηση της 14ης άσκησης θα βρίσκεται σε ένα αρχείο με όνομα: `Koulis_Nikos_Askisi_14.c`

Το σύνολο των αρχείων με τις απαντήσεις των ασκήσεων θα πρέπει να περιέχεται σε συμπιεσμένο αρχείο (`.rar` ή `.zip`) βάσει της ονοματολογίας: `Epitheto_Onoma.rar` ή `Epitheto_Onoma.zip`.

Π.χ. για τον προαναφερθέντα φοιτητή, το αρχείο αυτό θα είναι το: `Koulis_Nikos.rar` ή `Koulis_Nikos.zip`.

Ασκήσεις που θα παραδοθούν, αλλά **δεν θα ακολουθούν τους παραπάνω κανόνες ονοματολογίας**, **θα αγνοηθούν**.

Άσκηση 1: Συμπληρώστε τα παρακάτω κενά, ώστε το πρόγραμμα να εμφανίζει την ακόλουθη έξοδο:

```
21
 21
15
25%
A
 a
10
77
077
63
```

```c
#include <stdio.h>
int main()
{
	int x = 21, y = 0xa, z = 077;
	printf("_________\n", x);
	printf("_________\n", x);
	printf("_________\n", x);
	printf("_________\n", x);
	printf("_________\n", y);
	printf("_________\n", y);
	printf("_________\n", y);
	printf("_________\n", z);
	printf("_________\n", z);
	printf("_________\n", z);
	return 0;
}
```

Άσκηση 2: Να γραφεί ένα πρόγραμμα το οποίο να διαβάζει ακεραίους και να τους αποθηκεύει σε έναν πίνακα 100 θέσεων με τον περιορισμό ένας εισαγόμενος αριθμός να αποθηκεύεται στον πίνακα μόνο αν είναι μικρότερος από τον αριθμό που εισήχθηκε τελευταίος.

Άσκηση 3: Να γραφεί ένα πρόγραμμα που θα διαβάζει συνεχώς ακεραίους, έως ότου δοθεί ο αριθμός `0`. Στη συνέχεια, το πρόγραμμα θα εμφανίζει στην οθόνη τα ακόλουθα: πλήθος αρνητικών ακεραίων, πλήθος θετικών ακεραίων, άθροισμα αρνητικών ακεραίων, άθροισμα θετικών ακεραίων, άθροισμα περιττών ακεραίων και άθροισμα άρτιων ακεραίων *(ο αριθμός 0 δεν θεωρείται ούτε θετικός ούτε αρνητικός).*

Άσκηση 4: Να γραφεί ένα πρόγραμμα το οποίο να διαβάζει τρεις ακεραίους και την επιλογή του χρήστη και να χρησιμοποιεί την εντολή **`switch`** για να υποστηρίξει τρεις περιπτώσεις. Αν η επιλογή είναι 1, το πρόγραμμα να ελέγχει αν οι ακέραιοι είναι διαφορετικοί και να εμφανίζει ανάλογο μήνυμα. Αν είναι 2, να ελέγχει αν υπάρχουν δύο ακέραιοι που να είναι ίσοι και αν είναι 3, να εμφανίζει πόσοι από αυτούς ανήκουν στο [-3, 3].

Άσκηση 5: Πόσοι είναι οι όροι που πρέπει να προσθέσουμε από τις παρακάτω αριθμητικές σειρές (`S1` και `S2`), ώστε το άθροισμα των όρων της `S2` να γίνει κατά 100 μεγαλύτερο από το άθροισμα των αντίστοιχων όρων της `S1`;

`S1 = { 4, 10, 16, 22, ... }`
`S2 = { 1,  8, 15, 22, ... }`

Χρησιμοποιήστε **`do-while`** βρόχο.

Άσκηση 6: Να γραφεί ένα πρόγραμμα το οποίο να διαβάζει 100 πραγματικούς αριθμούς και να βρίσκει και να εμφανίζει τις δύο μικρότερες διαφορετικές τιμές.

Άσκηση 7: Να γραφεί ένα πρόγραμμα το οποίο με χρήση της συνάρτησης `rand()` να θέτει τυχαίες τιμές σε έναν πίνακα ακεραίων 5×4 με τον περιορισμό το άθροισμα κάθε γραμμής του να είναι ίσο με 100.

Άσκηση 8: Να γραφεί ένα πρόγραμμα το οποίο με χρήση δύο (το πολύ) επαναληπτικών βρόχων θα υπολογίζει και θα εκτυπώνει στην οθόνη το άθροισμα των ακεραίων:
`100 + 101 +…+ 105 + 200 + 201 + 202 + … + 205 +…+ 900 + 901 + 902 +…+ 905`

Άσκηση 9: Να γραφεί ένα πρόγραμμα το οποίο θα ζητά από τον χρήστη ακέραιους αριθμούς, τους οποίους θα αποθηκεύει σε έναν πίνακα ακεραίων 20 θέσεων. Η εισαγωγή των αριθμών θα σταματάει αν ο χρήστης εισάγει την τιμή `-1` ή αν δεν υπάρχει άλλη διαθέσιμη θέση αποθήκευσης στον πίνακα. Στη συνέχεια, το πρόγραμμα θα υπολογίζει τον μέσο όρο των τιμών του πίνακα και θα τον εμφανίζει στην οθόνη.

Άσκηση 10: Να γραφεί ένα πρόγραμμα το οποίο αρχικά να ζητά από τον χρήστη να εισάγει 9 ακέραιους, τους οποίους και θα αποθηκεύει σε έναν διδιάστατο πίνακα 3×3 (π.χ. `a[3][3]`) γεμίζοντάς τον σειριακά και ανά γραμμή. Στη συνέχεια, το πρόγραμμα:
Α) να εμφανίζει στην οθόνη τον πίνακα (με μορφή διδιάστατου πίνακα της άλγεβρας).
Β) να υπολογίζει και να εμφανίζει στην οθόνη σε ποια θέση (γραμμή/στήλη) του πίνακα βρίσκεται το ελάχιστο στοιχείο του.
Γ) να υπολογίζει και να εμφανίζει στην οθόνη το μέγιστο στοιχείο της κύριας διαγωνίου του.

Άσκηση 11: Να γραφεί ένα πρόγραμμα το οποίο, με χρήση της συνάρτησης `rand()`, αρχικά να εξομοιώνει 1000 συνεχόμενες τυχαίες ρίψεις ενός ζαριού (πιθανό αποτέλεσμα κάθε ρίψης: ακέραιος αριθμός από το 1 έως και το 6). Στη συνέχεια, τα 1000 αυτά αποτελέσματα να

αποθηκεύονται σε έναν πίνακα ακεραίων (έστω `zari[1000]`). Τέλος, το πρόγραμμα να υπολογίζει και να εμφανίζει στην οθόνη:
α) τον συνολικό αριθμό των ρίψεων με περιττό αποτέλεσμα.
β) τον συνολικό αριθμό των ρίψεων με αποτέλεσμα μικρότερο ή ίσο του 2.
γ) τον συνολικό αριθμό των συνεχόμενων ρίψεων με το ίδιο αποτέλεσμα.

Άσκηση 12: Να γραφεί ένα πρόγραμμα το οποίο να θέτει τυχαίες τιμές σε έναν πίνακα ακεραίων 10×10 με τον περιορισμό οι τιμές των στοιχείων κάτω από τη δευτερεύουσα διαγώνιο να είναι μικρότερες από τις τιμές των στοιχείων της δευτερεύουσας διαγωνίου, καθώς και οι τιμές των στοιχείων πάνω από τη δευτερεύουσα διαγώνιο να είναι μεγαλύτερες από τις τιμές των στοιχείων της δευτερεύουσας διαγωνίου. Το πρόγραμμα να εμφανίζει τον πίνακα στην οθόνη.

Άσκηση 13: Να γραφεί ένα πρόγραμμα το οποίο, με χρήση της συνάρτησης `rand()`, να αρχικοποιεί με μία τυχαία ακολουθία μονοψήφιων θετικών ακεραίων (δηλ. ακέραιες τιμές που ανήκουν στο διάστημα [0,9]) όλα τα στοιχεία ενός πίνακα ακεραίων 25 θέσεων και να τον εμφανίζει στην οθόνη, με τη μορφή αλγεβρικού πίνακα. Θεωρήστε ότι ο πίνακας θα είναι διδιάστατος 5×5 (π.χ. `int a[5][5]`). Στη συνέχεια το πρόγραμμα να αντικαθιστά κάθε στοιχείο του πίνακα με το αντίστοιχο παραγοντικό του αθροίσματος των δεικτών θέσης του κάθε στοιχείου. (π.χ. το στοιχείο `a[1][2]` να παίρνει την τιμή (1+2)! = 3!, το στοιχείο `a[4][3]` να παίρνει την τιμή (4+3)! = 7! κτλ), εκτός από τα στοιχεία της δευτερεύουσας διαγωνίου του (τα οποία να τα θέτει ίσα με μηδέν) και να τον εμφανίζει ξανά στην οθόνη.

Άσκηση 14: Να γραφεί ένα πρόγραμμα το οποίο να διαβάζει έναν ακέραιο (π.χ. Ν) και να εμφανίζει το αποτέλεσμα της παράστασης: $2^2+4^2+6^2+ \ldots +(2\times N)^2$. Το πρόγραμμα να υποχρεώνει τον χρήστη να εισάγει έναν θετικό ακέραιο μικρότερο του 20. Να χρησιμοποιήσετε έναν μόνο `for` βρόχο.

Άσκηση 15: Δημιουργήστε μία συνάρτηση η οποία θα υπολογίζει τον μέγιστο από 4 πραγματικούς αριθμούς (μεταβλητές τύπου `double`). Στη συνέχεια, να γραφεί ένα πρόγραμμα το οποίο θα ζητά από τον χρήστη να εισάγει 4 πραγματικούς αριθμούς και - με χρήση της παραπάνω συνάρτησης - θα εμφανίζει στην οθόνη τον μέγιστο από αυτούς.

Άσκηση 16: Δημιουργήστε μία `void` συνάρτηση που να δέχεται σαν παραμέτρους έναν `3×4` διδιάστατο πίνακα, τον αριθμό μίας γραμμής του και τον αριθμό μίας στήλης του και να επιστρέφει μέσω κατάλληλων παραμέτρων το μεγαλύτερο στοιχείο της γραμμής και το μικρότερο στοιχείο της στήλης. Να γραφεί ένα πρόγραμμα το οποίο να διαβάζει 12 ακεραίους και να τους αποθηκεύει σε έναν διδιάστατο πίνακα `3×4`. Στη συνέχεια, να διαβάζει τον αριθμό μίας γραμμής και μίας στήλης του και να εμφανίζει το μεγαλύτερο στοιχείο της γραμμής και το μικρότερο στοιχείο της στήλης με χρήση της συνάρτησης.

Άσκηση 17: Δημιουργήστε μία συνάρτηση η οποία θα δέχεται σαν παραμέτρους τρεις δείκτες σε `double` και θα επιστρέφει έναν δείκτη (σε `double`), στον `double` που έχει τη μικρότερη τιμή. Στη συνέχεια, να γραφεί ένα πρόγραμμα το οποίο θα διαβάζει τρεις δεκαδικούς αριθμούς και θα εμφανίζει τον μικρότερο από αυτούς με χρήση της παραπάνω συνάρτησης.

Άσκηση 18: Να γραφεί ένα πρόγραμμα το οποίο, με χρήση της συνάρτησης `rand()`, αρχικά θα εξομοιώνει 100 συνεχόμενες τυχαίες ρίψεις της μπίλιας μίας ρουλέτας (πιθανό αποτέλεσμα κάθε ρίψης: ακέραιος αριθμός από το 0 έως και το 36). Τα 100 αυτά αποτελέσματα θα αποθηκεύονται σε έναν μονοδιάστατο πίνακα ακεραίων (έστω `rouleta[100]`).
Θεωρήστε ότι ο παίχτης της ρουλέτας διαθέτει 200 μάρκες ίσης αξίας και ποντάρει συνεχώς και πάντοτε (δηλ. και στις 100 συνεχόμενες ρίψεις) μία μάρκα στο μηδέν (0) και ταυτόχρονα μία μάρκα στους περιττούς αριθμούς. Το κέρδος στην περίπτωση εμφάνισης του μηδενός (0) είναι: (αξία μάρκας)×36 ενώ το κέρδος στην περίπτωση εμφάνισης περιττού αριθμού είναι: (αξία μάρκας)×2.

Βάσει των παραπάνω, το πρόγραμμα να εμφανίζει στην οθόνη τη συχνότητα εμφάνισης των άρτιων αποτελεσμάτων, τη συχνότητα εμφάνισης των περιττών αποτελεσμάτων, τη συχνότητα εμφάνισης του μηδενός (0) και το συμπέρασμα για το πόσες μάρκες συνολικά κέρδισε ή έχασε ο παίκτης.

```
Π.χ. 1: ------------------------------------------------------------------
Emfanish artioy apotelsmatos: 53 fores
Emfanish perittoy apotelesmatos: 44 fores
Emfanish toy mhdenos (0): 3 fores
O PAIXTHS SYNOLIKA EINAI XAMENOS KATA: 4 MARKES

Π.χ. 2: ------------------------------------------------------------------
Emfanish artioy apotelsmatos: 46 fores
Emfanish perittoy apotelesmatos: 50 fores
Emfanish toy mhdenos (0): 4 fores
O PAIXTHS SYNOLIKA EINAI KERDISMENOS KATA: 44 MARKES
```

Άσκηση 19: Να γραφεί ένα πρόγραμμα το οποίο να αρχικοποιεί με χρήση της συνάρτησης `rand()` όλα τα στοιχεία δύο διδιάστατων (έναν 2×3 και έναν 3×2) πινάκων ακεραίων με τυχαίες τιμές που ανήκουν στο διάστημα [-9, 9] και να εμφανίζει τους δύο πίνακες στην οθόνη υπό τη μορφή αλγεβρικών πινάκων. Στη συνέχεια το πρόγραμμα να αποθηκεύει σε έναν τρίτο πίνακα (έστω `int` c [2][2]) το αντίστοιχο γινόμενο των πινάκων a και b (c = a × b), βάσει του πολλαπλασιασμού πινάκων – κατά τα γνωστά από την άλγεβρα – και το τελικό αποτέλεσμα να το εμφανίζει στην οθόνη με τη μορφή αλγεβρικού 2×2 πίνακα.

Σχόλια: Θυμίζουμε από τη γραμμική άλγεβρα ότι κατά τον πολλαπλασιασμό πινάκων τα στοιχεία του τελικού πίνακα προκύπτουν πολλαπλασιάζοντας διαδοχικά τα στοιχεία των γραμμών του 1ου πίνακα με τα στοιχεία των στηλών του 2ου. Συνεπώς, κατά τον πολλαπλασιασμό ενός N×M με έναν M×N πίνακα προκύπτει πίνακας διάστασης N×N. Για παράδειγμα, αν θεωρήσουμε τους παρακάτω πίνακες a (2×3) και b (3×2), όπου:

$$a = \begin{bmatrix} 1 & -1 & 1 \\ 0 & 2 & 1 \end{bmatrix} \quad \text{και} \quad b = \begin{bmatrix} 1 & 0 \\ 2 & -2 \\ 2 & 3 \end{bmatrix}$$

τότε, ο πίνακας c = a×b θα έχει διάσταση 2×2 και θα είναι ο εξής:

$$c = \begin{bmatrix} 1\times1+(-1)\times2+1\times2 & 1\times0+(-1)\times(-2)+1\times3 \\ 0\times1+2\times2+1\times2 & 0\times0+2\times(-2)+1\times3 \end{bmatrix} = \begin{bmatrix} 1 & 5 \\ 6 & -1 \end{bmatrix}$$

Παρατηρούμε δηλαδή ότι $c_{ij} = \sum_{k=1}^{M} a_{ik} \times b_{kj}$. Βέβαια, προγραμματιστικά, το k στο παραπάνω άθροισμα έχει τιμές από μηδέν (0) έως M-1 και όχι από ένα (1) έως M που ισχύει στην άλγεβρα πινάκων.

Άσκηση 20: Δημιουργήστε μία συνάρτηση που να προσομοιώνει την κλήρωση των αριθμών του τυχερού παιχνιδιού Τζόκερ. Η συνάρτηση, θα πρέπει να χρησιμοποιεί τη συνάρτηση `rand()` και κατάλληλες παραμέτρους, ώστε να επιστρέφει 5 ακεραίους που ανήκουν στο [1, 45] και έναν ακέραιο που ανήκει στο [1, 20] (που είναι ο αριθμός Τζόκερ). Στη συνέχεια, να γραφεί ένα πρόγραμμα που, με χρήση της συνάρτησης, αρχικά θα δημιουργεί 1000 τυχαίες στήλες του παιχνιδιού (κάθε στήλη περιέχει πέντε αριθμούς στο [1, 45] και έναν αριθμό Τζόκερ), στη συνέχεια θα εξομοιώνει την κλήρωση των 5 τυχερών αριθμών και του τυχερού αριθμού Τζόκερ και, τέλος, θα ελέγχει πόσες από τις 1000 στήλες ανήκουν σε κατηγορία που κερδίζει, αναγράφοντας τα κατάλληλα στατιστικά μηνύματα.
Σημειώνεται, ότι στο τυχερό παιχνίδι Τζόκερ οι οκτώ κατηγορίες που κερδίζουν βάσει των αντίστοιχων προβλέψεων είναι οι εξής:

5+1 (πέντε αριθμοί και ο αριθμός Τζόκερ)
5+0 (πέντε αριθμοί)
4+1 (τέσσερις αριθμοί και ο αριθμός Τζόκερ)
4+0 (τέσσερις αριθμοί)
3+1 (τρεις αριθμοί και ο αριθμός Τζόκερ)
3+0 (τρεις αριθμοί)
2+1 (δύο αριθμοί και ο αριθμός Τζόκερ)
1+1 (ένας αριθμός και ο αριθμός Τζόκερ)

```
Π.χ. : ------------------------------------------------------------------
TYXEROI ARI8MOI: 3, 6, 14, 39, 44
ARI8MOS TZOKER: 13

Stis 1000 sthles poy paixthkan, bre8hkan ta parakatw apotelesmata:
5+1: 0 sthles
5+0: 0 sthles
4+1: 0 sthles
4+0: 2 sthles
3+1: 3 sthles
3+0: 14 sthles
2+1: 55 sthles
1+1: 67 sthles
...kai 859 sthles den kerdizoun tipota!!!
```

Άσκηση 21: Δημιουργήστε μία συνάρτηση απλής κωδικοποίησης, η οποία θα δέχεται σαν παραμέτρους έναν χαρακτήρα (π.χ. **char** c) και έναν ακέραιο (π.χ. **int** x) και θα επιστρέφει τον χαρακτήρα που έχει μεγαλύτερη κατά x ASCII τιμή από τον χαρακτήρα c. Στη συνέχεια, να γραφεί ένα πρόγραμμα το οποίο θα διαβάζει έναν χαρακτήρα και έναν ακέραιο και θα εμφανίζει τον κωδικοποιημένο χαρακτήρα με χρήση της παραπάνω συνάρτησης.

Άσκηση 22: Να γραφεί ένα πρόγραμμα το οποίο, με χρήση της συνάρτησης `rand()`, να γεμίζει με τυχαίες ακέραιες τιμές μεταξύ 0 και M, έναν διδιάστατο τετραγωνικό πίνακα `r` διάστασης N×N. Θεωρήστε ότι τα M και N είναι θετικές ακέραιες σταθερές του προγράμματος. Στη συνέχεια, το πρόγραμμα να υπολογίζει και να εκτυπώνει τη συχνότητα εμφάνισης κάθε τιμής στον πίνακα `r`, δηλαδή πόσες φορές κάθε τιμή εμφανίζεται στον πίνακα. Θα χρειαστείτε έναν δεύτερο μονοδιάστατο πίνακα, έστω `freq`, με M+1 στοιχεία.

Άσκηση 23: Να γραφεί ένα πρόγραμμα το οποίο, με χρήση της συνάρτησης `rand()`, θα παράγει συνεχώς ακεραίους με τυχαίες τιμές που ανήκουν στο διάστημα [1, 10] και κάθε φορά θα ρωτάει τον χρήστη να μαντέψει αν ο ακέραιος που μόλις παρήγαγε είναι περιττός ή άρτιος. Ο χρήστης θα απαντάει πληκτρολογώντας 1 (αν πιστεύει ότι είναι περιττός) ή 2 (αν πιστεύει ότι είναι άρτιος), αντίστοιχα. Αφού ο χρήστης απαντήσει, το πρόγραμμα θα πρέπει να εμφανίζει στην οθόνη τον ακέραιο που είχε παραχθεί καθώς και κατάλληλο μήνυμα («Κερδίσατε» ή «Χάσατε», αντίστοιχα). Σε περίπτωση που ο χρήστης μάντεψε σωστά, το πρόγραμμα θα πρέπει να συνεχίζει ακολουθώντας την ίδια λογική. Το πρόγραμμα θα τερματίζει όταν ο χρήστης μαντέψει λάθος και τότε θα πρέπει να εκτυπώνεται στην οθόνη ο αριθμός των σωστών απαντήσεων που έχει δώσει συνολικά ο χρήστης μέχρι να χάσει.

Άσκηση 24: Δημιουργήστε μία συνάρτηση που θα δέχεται σαν παραμέτρους έναν πίνακα από **float** και τη διάσταση του πίνακα και θα επιστρέφει έναν δείκτη (σε **float**) στο τελευταίο στοιχείο του πίνακα. Στη συνέχεια, να γραφεί ένα πρόγραμμα το οποίο θα ζητά από τον χρήστη 5 δεκαδικούς αριθμούς, θα τους αποθηκεύει σε έναν πίνακα από **float** και θα εμφανίζει στην οθόνη το τελευταίο στοιχείο του πίνακα με χρήση της παραπάνω συνάρτησης.

Άσκηση 25: Οι δυνατοί συνδυασμοί των n στοιχείων ανά k δίνονται από τον τύπο: $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ όπου n, k θετικοί ακέραιοι και $n \geq k$.

Δημιουργήστε μία συνάρτηση που θα υπολογίζει το παραγοντικό ενός μη αρνητικού ακέραιου αριθμού. Στη συνέχεια, να γραφεί ένα πρόγραμμα το οποίο θα ζητά από τον χρήστη να εισάγει δύο ακέραιους αριθμούς (n και k) και θα ελέγχει συνεχώς αν οι δύο ακέραιοι είναι θετικοί και αν ο πρώτος (n) είναι μεγαλύτερος ή ίσος από τον δεύτερο (k). Αν η παραπάνω συνθήκη είναι αληθής, τότε το πρόγραμμα θα υπολογίζει τους συνδυασμούς των n στοιχείων ανά k με χρήση της συνάρτησης και στη συνέχεια θα τερματίζει. Σε διαφορετική περίπτωση (αν δηλ. η συνθήκη είναι ψευδής) το πρόγραμμα να μην τερματίζει και να συνεχίζει επίμονα να ζητάει από τον χρήστη δύο ακέραιους αριθμούς (n και k), έως ότου ικανοποιηθεί η συνθήκη (και τότε - προφανώς - να υπολογίζει τους δυνατούς συνδυασμούς των n στοιχείων ανά k).

Άσκηση 26: Να γραφεί ένα πρόγραμμα το οποίο να προσομοιώνει ένα αυτόματο σύστημα έκδοσης εισιτηρίων για κάποιο δρομολόγιο λεωφορείου του ΚΤΕΛ. Θεωρήστε ότι το λεωφορείο διαθέτει 48 καθίσματα (12 σειρές με 4 θέσεις σε κάθε σειρά). Το πρόγραμμα να δίνει τη δυνατότητα στον χρήστη να επιλέξει τις παρακάτω λειτουργίες, οι οποίες θα εκτελούνται με κλήση των αντίστοιχων συναρτήσεων που θα πρέπει να έχετε υλοποιήσει.

1) Κράτηση-Αγορά εισιτηρίου. Το πρόγραμμα να εμφανίζει στον χρήστη ένα διάγραμμα (με τη μορφή πίνακα 12×4) με τις ελεύθερες θέσεις και τις θέσεις που έχουν ήδη κρατηθεί και να του δίνει τη δυνατότητα να κάνει κράτηση μίας θέσης από τις διαθέσιμες. Σε περίπτωση που ο χρήστης επιλέξει μία θέση που είναι ήδη κρατημένη, να εμφανίζεται κατάλληλο μήνυμα στην οθόνη και η κράτηση/αγορά να μην πραγματοποιείται. Σε περίπτωση που ο χρήστης επιλέξει μία ελεύθερη θέση, να πραγματοποιείται η κράτηση και ταυτόχρονα να γίνεται και η αγορά του εισιτηρίου. Το κόστος του εισιτηρίου είναι 15€.

2) Εισπράξεις – Πλάνο. Εμφάνιση του συνολικού ποσού που έχει εισπραχθεί για το συγκεκριμένο δρομολόγιο του ΚΤΕΛ και ενός διαγράμματος του λεωφορείου (με τις ελεύθερες θέσεις και τις θέσεις που έχουν κρατηθεί έως εκείνη τη στιγμή).

3) Ακύρωση εισιτηρίου. Το πρόγραμμα να διαβάζει τη σειρά και τη θέση μιας κράτησης και να την ακυρώνει. Το ποσό επιστροφής είναι 12€.

4) Τερματισμός προγράμματος.

Άσκηση 27: Δημιουργήστε μία συνάρτηση που να δέχεται σαν παράμετρο έναν διδιάστατο πίνακα και έναν αριθμό τύπου **double** και να επιστρέφει έναν δείκτη στη γραμμή όπου αυτός ο αριθμός εμφανίζεται τις περισσότερες φορές. Αν ο αριθμός δεν υπάρχει στον πίνακα, η συνάρτηση να επιστρέφει NULL. Να γραφεί ένα πρόγραμμα το οποίο να θέτει τυχαίες τιμές σε έναν 5x5 πίνακα τύπου **double**, να διαβάζει έναν αριθμό τύπου **double** και να χρησιμοποιεί τη συνάρτηση για να εμφανίσει τα στοιχεία της γραμμής στην οποία περιέχεται τις περισσότερες φορές.

Άσκηση 28: Να γραφεί ένα πρόγραμμα το οποίο να διαβάζει ένα αλφαριθμητικό μέχρι 100 χαρακτήρες και να το αποθηκεύει σε έναν πίνακα χαρακτήρων. Στη συνέχεια, να απομακρύνει από τον πίνακα όλους τους χαρακτήρες που δεν είναι γράμματα καθώς και τις πολλαπλές εμφανίσεις του ίδιου γράμματος, και να εμφανίζει το νέο αλφαριθμητικό. Για παράδειγμα, αν στον πίνακα έχει αποθηκευτεί το jA*8%cr1^c, το πρόγραμμα να μεταβάλει το περιεχόμενό του σε jAcr.

(Σημείωση: Να μην χρησιμοποιήσετε βοηθητικό πίνακα για την αποθήκευση των χαρακτήρων που δεν διαγράφονται).

Άσκηση 29: Θεωρήστε ότι κάθε πεζό γράμμα του αγγλικού αλφαβήτου έχει μία τιμή, ως εξής. Τα γράμματα `'a'` έως και `'i'` αντιστοιχίζονται σε μονάδες με το `'a'` να έχει την τιμή `1` και το `'i'` την τιμή `9`. Τα γράμματα `'j'` έως και `'r'` αντιστοιχίζονται σε δεκάδες με το `'j'` να έχει την τιμή `10` και το `'r'` την τιμή `90`. Τα υπόλοιπα αντιστοιχίζονται σε εκατοντάδες με το `'s'` να έχει την τιμή `100` και το `'z'` την τιμή `800`. Όλοι οι υπόλοιπο χαρακτήρες έχουν την τιμή μηδέν.

Να γραφεί ένα πρόγραμμα το οποίο να διαβάζει συνεχώς αλφαριθμητικά μέχρι 100 χαρακτήρες και πριν τερματίσει να εμφανίσει τα αλφαριθμητικά με τη μεγαλύτερη και τη μικρότερη τιμή, και ποιες είναι αυτές. Για παράδειγμα, η τιμή του αλφαριθμητικού `cky` είναι `723`, ενώ η τιμή του `A9m` είναι `40`. Αν περισσότερα από ένα αλφαριθμητικά έχουν την ίδια μεγαλύτερη ή μικρότερη τιμή, το πρόγραμμα να εμφανίζει το τελευταίο που εισήγαγε ο χρήστης. Αν ο χρήστης εισάγει «τρία αστέρια» ως αλφαριθμητικό `***`, η εισαγωγή των αλφαριθμητικών να τερματίζει.

Άσκηση 30: Να γραφεί ένα πρόγραμμα το οποίο να διαβάζει 20 αλφαριθμητικά μέχρι 100 χαρακτήρες και να τα αποθηκεύει σε έναν διδιάστατο πίνακα. Στη συνέχεια, το πρόγραμμα:

α. να διαβάζει τον αριθμό μίας στήλης και να εμφανίζει τις λέξεις που περιέχονται σε αυτή (ακόμα και αυτές που αποτελούνται από ένα χαρακτήρα). Σε ένα παράδειγμα με 5 αλφαριθμητικά, αν ο χρήστης εισάγει τα:

```
abcdegm
abcdfpoloo
abcd
abc
abcdkmop
```

και αριθμό στήλης το `6`, το πρόγραμμα να εμφανίζει:

```
word_1: gp
word_2: m
```

β. να διαβάζει τον αριθμό μίας γραμμής εκτός της πρώτης και αν οι διαφορετικοί πεζοί χαρακτήρες που περιέχονται σε αυτή είναι λιγότεροι από 5 να αντιμεταθέτει το αλφαριθμητικό αυτής της γραμμής με το αλφαριθμητικό της πρώτης γραμμής.

Άσκηση 31: Να γραφεί ένα πρόγραμμα το οποίο να διαβάζει τα ονόματα 20 βιβλιοπωλείων, 500 τίτλους βιβλίων και τον αριθμό των αντιτύπων που έχει για κάθε τίτλο το κάθε βιβλιοπωλείο (θεωρήστε ότι όλα τα ονόματα είναι μέχρι 100 χαρακτήρες το καθένα). Το πρόγραμμα να χρησιμοποιεί κατάλληλους πίνακες για την αποθήκευση των δεδομένων. Στη συνέχεια, το πρόγραμμα να διαβάζει τον τίτλο ενός βιβλίου και να εμφανίζει τα ονόματα των βιβλιοπωλείων στα οποία υπάρχουν διαθέσιμα αντίτυπα και πόσα είναι αυτά. Τέλος, να διαβάζει το όνομα ενός βιβλιοπωλείου και να εμφανίζει το πλήθος των βιβλίων που διαθέτει.

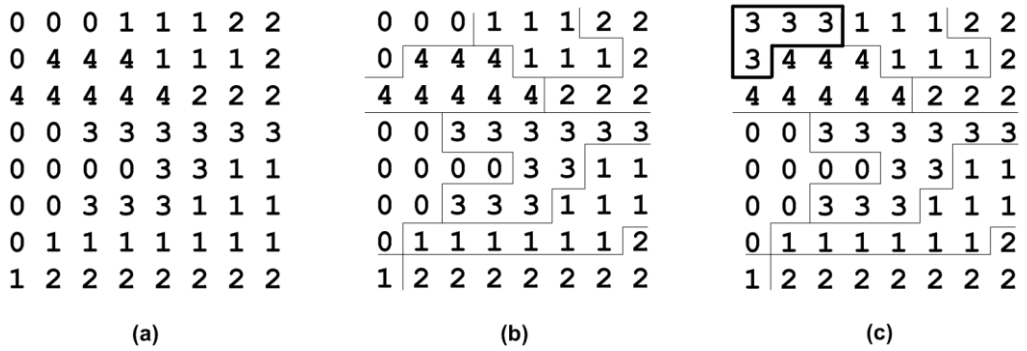
Άσκηση 32: Δημιουργήστε τη συνάρτηση `power(double m, int n)`, η οποία να επιστρέφει το m^n, υπολογίζοντάς το βάσει του αναδρομικού τύπου: $m^n = m \times m^{n-1}$ Να γραφεί ένα πρόγραμμα, το οποίο να διαβάζει έναν πραγματικό αριθμό m και έναν ακέραιο n και να εμφανίζει το αποτέλεσμα της πράξης m^n με χρήση της συνάρτησης.

Άσκηση 33: Δίνεται η αναδρομική σχέση:

$$\mathrm{K}(n) = \frac{\mathrm{K}(n-1) \times n!}{\left(\mathrm{K}(n-1)+1\right)^2}, \text{ για } n > 0 \quad \text{και} \quad \mathrm{K}(0) = 1$$

Να γραφεί ένα πρόγραμμα το οποίο να διαβάζει έναν θετικό ακέραιο (π.χ. n) και να χρησιμοποιεί μία αναδρομική συνάρτηση για να εμφανίσει τον K(n) βάσει της παραπάνω σχέσης.

Άσκηση 34: Η συγκεκριμένη εφαρμογή αποτελεί την υλοποίηση ενός κλασικού αλγορίθμου (που έστω ότι ονομάζεται «αλγόριθμος του Κούλη»), τον οποίο χρησιμοποιούν προγράμματα επεξεργασίας εικόνας (π.χ. photoshop) για να χρωματίσουν γειτονικά τμήματα μίας εικόνας με ένα νέο χρώμα. Ας υποθέσουμε ότι ο διδιάστατος 8×8 πίνακας του πρώτου σχήματος (a) αντιπροσωπεύει τα στοιχεία (pixels) μίας εικόνας, όπου το 0 αντιστοιχεί στο μαύρο χρώμα, 1: λευκό, 2: κόκκινο, 3: πράσινο και 4: μπλε. Θεωρούμε ότι ένα στοιχείο συνδέεται χρωματικά με κάποιο άλλο, μόνο αν είναι γειτονικό του και έχει το ίδιο χρώμα. Για παράδειγμα, οι ομοιόμορφα χρωματισμένες περιοχές διαχωρίζονται και απεικονίζονται στο δεύτερο σχήμα (b).



Για την υλοποίηση του αλγορίθμου του Κούλη, να δημιουργήσετε την αναδρομική συνάρτηση **`koulis()`**, η οποία να αλλάζει το χρώμα (π.χ. **`c`**) ενός στοιχείου σε μία τυχαία θέση (π.χ. **`i,j`**) σε ένα νέο χρώμα (π.χ. **`nc`**) και στη συνέχεια να αλλάζει το χρώμα και των γειτονικών του στοιχείων (δηλ., των στοιχείων που βρίσκονται αριστερά, δεξιά, πάνω και κάτω από αυτό) των οποίων το χρώμα είναι επίσης **`c`**. Προσέξτε ότι αυτή η διαδικασία πρέπει να συνεχίσει αναδρομικά και με τα γειτονικά στοιχεία αυτών που αλλάζει το χρώμα, μέχρι να μην υπάρχει άλλο στοιχείο να ελεγχθεί.
Για παράδειγμα, αν επιλέξουμε να αλλάξουμε το χρώμα του στοιχείου που βρίσκεται στη θέση **`0,0`** από μαύρο (π.χ. **`0`**) σε πράσινο (π.χ. **`3`**), τότε αλλάζει το χρώμα των τεσσάρων στοιχείων της πάνω-αριστερά περιοχής, όπως φαίνεται στο τρίτο σχήμα (c).

Να γραφεί ένα πρόγραμμα το οποίο να θέτει τυχαίες τιμές στο `[0, 4]` στα στοιχεία ενός 8×8 πίνακα ακεραίων. Στη συνέχεια, το πρόγραμμα να διαβάζει τη θέση ενός στοιχείου (π.χ. `i,j`) και ένα νέο χρώμα και να χρησιμοποιεί τη συνάρτηση `koulis()` για να αλλάξει το χρώμα των γειτονικών του στοιχείων με το νέο αυτό χρώμα.

Άσκηση 35: Να γραφεί ένα πρόγραμμα το οποίο (μόνο με χρήση τελεστών και χωρίς τις εντολές **`if`**/**`for`**/**`while`** κτλ) να διαβάζει έναν θετικό ακέραιο και να εμφανίζει τον πλησιέστερο μεγαλύτερο αριθμό που να είναι δύναμη του 2. Για παράδειγμα, αν ο χρήστης εισάγει 35, το πρόγραμμα να εμφανίζει 64 (αφού $2^5 = 32 < 35 < 64 = 2^6$).