

## ΠΡΟΤΕΙΝΟΜΕΝΑ ΘΕΜΑΤΑ 3 στον Προγραμματισμό Υπολογιστών

### ΘΕΜΑ Α

**A1.** Να χαρακτηρίσετε τις προτάσεις που ακολουθούν, γράφοντας στο τετράδιό σας, δίπλα στο γράμμα που αντιστοιχεί σε κάθε πρόταση τη λέξη **Σωστό**, αν η πρόταση είναι σωστή ή τη λέξη **Λάθος**, αν η πρόταση είναι λανθασμένη.

**α.** Η δυαδική αναζήτηση βρίσκει το ζητούμενο πολύ πιο γρήγορα από ότι η σειριακή αναζήτηση.

**β.** Η μέθοδος λίστας **L.insert(0,2)** προσθέτει το στοιχείο 0, στη θέση 2 της λίστας L, μετακινώντας όλα τα στοιχεία από τη θέση 2 και μετά, κατά μία θέση.

**γ.** Η **divmod(x,y)** επιστρέφει το ακέραιο πηλίκο και το ακέραιο υπόλοιπο της διαίρεσης **x/y**.

**δ.** Η **Ουρά (Queue)** ανήκει στους **Σύνθετους Τύπους Δεδομένων**.

**ε.** Η μέθοδος **fin.tell()** επιστρέφει έναν ακέραιο που περιέχει την τρέχουσα θέση στο αρχείο fin, υπολογισμένη σε χαρακτήρες (bytes) από την αρχή του αρχείου.

**Μονάδες 10**

**A2.** Να γράψετε στο τετράδιό σας τους αριθμούς **1, 2, 3, 4, 5** από τη Στήλη **A** του παρακάτω πίνακα και δίπλα ένα από τα γράμματα **α, β, γ, δ, ε, στ** της Στήλης **B**, που δίνει τη σωστή αντιστοιχία. Σημειώνεται ότι ένα γράμμα από τη στήλη **B** θα περισσέψει.

ΣΤΗΛΗ Α	ΣΤΗΛΗ Β
1. range(1,10,1)	α. [10]
2. range(10,1,-1)	β. [ ]
3. range(10,-1,-1)	γ. [10,9,8,7,6,5,4,3,2]
4. range(10,0,-1)	δ. [1,2,3,4,5,6,7,8,9]
5. range(10,10,1)	ε. [10,9,8,7,6,5,4,3,2,1,0]
	στ. [10,9,8,7,6,5,4,3,2,1]

**Μονάδες 5**

**A3.** Να μεταφέρετε και να συμπληρώσετε στο τετράδιό σας τον παρακάτω πίνακα με τα αποτελέσματα των πράξεων μεταξύ τριών μεταβλητών X, Y, Z.

X	Y	Z	X>Y and not(Y<=Z)	X>2 and Y<4 or Z>=5
10	5	3		
2	5	6		
12	3	5		

**Μονάδες 6**

**A4.** Ο παρακάτω αλγόριθμος επιστρέφει τη θέση του στοιχείου *key* αν υπάρχει μέσα στη λίστα *array*, σε διαφορετική περίπτωση επιστρέφει  $-1$

```
def binarySearch( array, key ) :  
    first = 0  
    last = len(array) - 1  
    pos = (1)  
    while first <= last and pos (2) -1 :  
        mid = ( first + last ) / 2  
        if array[ mid ] == key :  
            pos = (3)  
        elif array[ mid ] (4) key :  
            first = mid + 1  
        else :  
            last = mid - 1  
    return pos
```

Στο τμήμα προγράμματος υπάρχουν υπογραμμισμένα κενά τα οποία έχουν αριθμηθεί. Να γράψετε στο τετράδιό σας τους αριθμούς 1, 2, 3 και 4 που αντιστοιχούν στα κενά του παραπάνω τμήματος προγράμματος και δίπλα σε κάθε αριθμό αυτό που πρέπει να συμπληρωθεί ώστε να υλοποιείται σωστά η δυαδική αναζήτηση που επιστρέφει τη θέση *pos* ενός στοιχείου *key* μέσα σε μία λίστα *array*.

**Μονάδες 4**

### **ΘΕΜΑ Β**

**B1.** Οι παρακάτω συναρτήσεις αφορούν την υλοποίηση της Ουράς. Να τις ξαναγράψετε στο τετράδιό σας συμπληρώνοντας τα κενά:

**def enqueue(queue, item) :**  
\_\_\_\_\_

**def dequeue(queue) :**  
\_\_\_\_\_

**def isEmpty(queue) :**  
\_\_\_\_\_

**def createQueue( ) :**  
\_\_\_\_\_

**Μονάδες 8**

**B2.**

Τι θα εμφανιστεί στην οθόνη του Η/Υ μετά την εκτέλεση των παρακάτω εντολών

```
A=createQueue()  
enqueue(A,10)  
enqueue(A,20)  
print dequeue(A)  
enqueue(A,40)  
if not isEmpty(A):  
    enqueue(A,87)  
    enqueue(A,-6)  
print dequeue(A)  
print dequeue(A)
```

**Μονάδες 6**

**B3.** Να υλοποιήσετε την δομή δεδομένων “ουρά” ως μια κλάση αξιοποιώντας τις τεχνικές του αντικειμενοστρεφούς προγραμματισμού. Ξαναγράψτε στο τετράδιό σας την παρακάτω κλάση συμπληρώνοντας τα κενά.

**class Queue :**

```
def __init__(self) :  
    self.items = []  
  
def enqueue(self, item) :  
    _____  
  
def dequeue(self) :  
    _____  
  
def isEmpty(self) :  
    _____
```

### Μονάδες 3

**B4.** Να ξαναγράψετε το τμήμα προγράμματος του B2 χρησιμοποιώντας μόνο την κλάση που δημιουργήσατε στο B3.

```
A=Queue()  
A.enqueue(10)  
_____(1)_____  
print _____(2)_____  
_____(3)_____  
if not _____(4)_____  
    _____(5)_____  
    _____(6)_____  
print _____(7)_____  
print _____(8)_____
```

### Μονάδες 8

### ΘΕΜΑ Γ

Η Εταιρεία ύδρευσης της περιοχής μας ΔΕΥΑΠ (το τελευταίο Π από το Python) χρεώνει κλιμακωτά τους πελάτες της ανάλογα με τα κυβικά μέτρα νερού που κατανάλωσαν, σύμφωνα με τον παρακάτω πίνακα:

Κυβικά μέτρα νερού	Τιμή ανά κυβικό μέτρο
Από 0 έως και 40	0,25 €
Πάνω από 40 έως και 90	0,30 €
Πάνω από 90	0,36 €

Δηλαδή αν κάποιος κατανάλωσε 50 κυβικά μέτρα νερού θα πληρώσει  $40 \times 0,25$  για τα πρώτα 40 και για τα υπόλοιπα 10 ,  $10 \times 0,30$ , Δηλαδή συνολικά:

Κόστος νερού =  $40 \times 0,25 + 10 \times 0,30 = 10 + 3 = 13$  €

Να γράψετε πρόγραμμα σε γλώσσα προγραμματισμού Python, το οποίο:

**Γ1.** Να περιέχει συνάρτηση βρες vres\_kostos η οποία να δέχεται σαν είσοδο τα κυβικά που κατανάλωσε ο πελάτης και να επιστρέφει το κόστος νερού σύμφωνα με τον παραπάνω πίνακα και με κλιμακωτή χρέωση.

**Μονάδες 5**

**Γ2.** Να διαβάζει το Επώνυμο και το όνομα του καταναλωτή και τα κυβικά που κατανάλωσε και να τα καταχωρεί στις λίστες EP,ON,KYB αντίστοιχα.

**Μονάδες 5**

**Γ3.** Η διαδικασία να επαναλαμβάνεται μέχρι να δοθεί για επώνυμο η λέξη “TELOS”.

**Μονάδες 2**

**Γ4.** Με τη βοήθεια της συνάρτησης του Γ1 να υπολογίζει και να εμφανίζει το κόστος νερού για κάθε ένα πελάτη. Να το καταχωρεί σε μία νέα λίστα ΚΟΣΤΟΣΝ.

**Μονάδες 4**

**Γ5.** Με τη βοήθεια τις bubbleSort για 4 λίστες να ταξινομήσετε τις λίστες ως προς το κόστος νερού με αύξουσα ταξινόμηση και να εμφανίσετε τα ονοματεπώνυμα των 3 πελατών με το μεγαλύτερο κόστος νερού καθώς επίσης και τα κυβικά μέτρα νερού που κατανάλωσαν.(Θεωρούμε ότι μας δόθηκαν τουλάχιστον 3 πελάτες)

**Μονάδες 6**

**Γ6.** Για κάθε πελάτη υπάρχει ένας επιπλέον φόρος που είναι για τη δημιουργία έργων ύδρευσης και αποχέτευσης. Ο φόρος αυτός είναι το 80% του κόστους του νερού. Να υπολογίσετε και να εμφανίσετε το φόρο έργων και το τελικό ποσό που πρέπει να πληρώσει ο κάθε πελάτης.

**Μονάδες 3**

#### **ΘΕΜΑ Δ**

Στους αγώνες άλματος με σκι ο κάθε αθλητής κάνει δύο άλματα. Για κάθε άλμα παίρνει μία βαθμολογία. Η τελική βαθμολογία προκύπτει από το άθροισμα των δύο βαθμολογιών. Ο νικητής του αγωνίσματος είναι αυτός που θα έχει τη μεγαλύτερη τελική βαθμολογία.

Να γράψετε πρόγραμμα σε Python το οποίο:

**Δ1.** Να διαβάζει το επώνυμο, το όνομα, τη βαθμολογία στο πρώτο άλμα, τη βαθμολογία στο δεύτερο άλμα και τη χώρα του αθλητή.

**Μονάδες 4**

**Δ2.** Η παραπάνω διαδικασία να επαναλαμβάνεται για 80 αθλητές.

**Μονάδες 2**

**Δ3.** Να καταχωρίζει τα στοιχεία στις λίστες EP,ON, ALMA1, ALMA2 και XWRA αντίστοιχα.

**Μονάδες 4**

**Δ4.** Για κάθε αθλητή να υπολογίζει και να εμφανίζει τη συνολική βαθμολογία του (αθροίζοντας τα δύο άλματα που έκανε) και να την καταχωρεί σε μία νέα λίστα SYN.

**Μονάδες 4**

**Δ5.** Να υπολογίζει και να εμφανίζει το άθροισμα της συνολικής βαθμολογίας όλων των αθλητών από τη ΓΕΡΜΑΝΙΑ

**Μονάδες 4**

**Δ6.** Να υπολογίζει και να εμφανίζει το επώνυμο και το όνομα του αθλητή με τη μεγαλύτερη συνολική επίδοση

**Μονάδες 4**

**Δ7.** Να υπολογίζει και να εμφανίζει το πλήθος των αθλητών που είχαν μεγαλύτερη βαθμολογία στο δεύτερο άλμα σε σχέση με το πρώτο άλμα.

**Μονάδες 3**

**Καλή επιτυχία!**