**ΟΔΗΓΙΕΣ**

* *Η εργασία γίνεται σε ομάδες 2 ατόμων.*
* *Η παράδοση της εργασίας είναι έως τις* ***20/05/2022****.*
* *Ο βαθμός της εργασίας μπορεί να κρατηθεί μέχρι και την εξεταστική του Σεπτεμβρίου 2022.*
* *Τα προγράμματά σας θα περιέχουν οπωσδήποτε σχόλια! Τα σχόλια αποτελούν μέρος του κώδικα και βαθμολογούνται.*
* *Η ΑΝΤΙΓΡΑΦΗ ΑΝΙΧΝΕΥΕΤΑΙ ΚΑΙ ΤΙΜΩΡΕΙΤΑΙ.*

**Α. (4 μονάδες)** Να αναπτυχθεί ένα πρόγραμμα το οποίο ορίζει 3 κλάσεις τις **PC**, **Lab** και **Test**.

Η 1η κλάση **PC** έχει τα παρακάτω πεδία, κατασκευαστές και μεθόδους:

**Πεδία**

* **processor**: Επεξεργαστής (string).
* **frequency**: Συχνότητα λειτουργίας σε GHz (δεκαδικός αριθμός)
* **RAM**: Μέγεθος μνήμης RAM σε GB (ακέραιος αριθμός)
* **freeSpace**: Ελεύθερος χώρος στον δίσκο σε GB (δεκαδικός αριθμός)
* **operational**: Δηλώνει αν ο Η/Υ είναι λειτουργικός (true/false)

**Κατασκευαστές**

* **χωρίς ορίσματα**: αρχικοποιεί όλα τα πεδία σε ουδέτερες τιμές ή κενά.
* **με ορίσματα**: έχει ως ορίσματα τιμές για όλα τα πεδία.

**Μέθοδοι**

* Συναρτήσεις **set** και **get** για όλα τα πεδία.
* **read()**: ζητάει από το χρήστη, ελέγχει και αποθηκεύει τα στοιχεία για το αντικείμενο που την καλεί.
* **toString()**: επιστρέφει σε μορφή αλφαριθμητικού όλα τα πεδία της κλάσης.

Η 2η κλάση **Lab** έχει τα παρακάτω πεδία, κατασκευαστές και μεθόδους:**Πεδία**

* **name**: όνομα του εργαστηρίου (string)
* **computers**: υπολογιστές που διαθέτει (ArrayList με αντικείμενα τύπου PC)

**Κατασκευαστής χωρίς ορίσματα**: αρχικοποιεί το ArrayList (this.computers = new ArrayList<>()).

**Μέθοδοι**

* Συναρτήσεις **set** και **get** για το name.
* **addComputer(PC npc)**: Προσθέτει τον υπολογιστή *npc* στο ArrayList *computers*.
* **int workingPCs()**: Τυπώνει όλα τα στοιχεία των Η/Υ που δε λειτουργούν και επιστρέφει τον αριθμό των Η/Υ που λειτουργούν χωρίς πρόβλημα.
* **boolean installApp(float size)**: Ελέγχει αν όλοι οι Η/Υ έχουν διαθέσιμο χώρο για την εγκατάσταση μίας εφαρμογής με μέγεθος *size* και αν ναι την εγκαθιστά. Δηλαδή μειώνει το πεδίο *freeSpace* όλων των υπολογιστών του εργαστηρίου κατά *size*.
* **pcStatus(int index, boolean status)**: Αλλάζει το πεδίο *operational* του υπολογιστή στη θέση *index* του ArrayList σε *status*.
* **boolean checkSpecs(float freq, int ram, float size)**: ελέγχει αν το εργαστήριο μπορεί να υποστηρίξει την εκτέλεση μίας εφαρμογής με τις παραπάνω προδιαγραφές και τυπώνει όλα τα στοιχεία των Η/Υ που δεν τις πληρούν.
* **toString()**: η οποία επιστρέφει σε μορφή αλφαριθμητικού όλα τα πεδία της κλάσης.

Η κλάση **Test**, θα έχει τη συνάρτηση main, στην οποία θα κάνετε τα παρακάτω:

* Δημιουργήστε ένα αντικείμενο τύπου Lab και εισάγετε τους παρακάτω Η/Υ:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **processor** | **frequency (GHz)** | **RAM**  **(GB)** | **freeSpace (GB)** | **operational** |
| Intel Core i3-12300 | 2.3 | 8 | 32.3 | true |
| Intel Core i5-12600 | 3.3 | 16 | 523 | true |
| AMD Ryzen 5 3600 | 3.6 | 4 | 10 | true |
| AMD Ryzen 7 5750 | 3.8 | 16 | 410 | true |

* Εκτυπώστε τον αριθμό Η/Υ του εργαστηρίου οι οποίοι λειτουργούν κανονικά.
* Προσπαθήστε να εγκαταστήσετε μία εφαρμογή με μέγεθος 3GB και τυπώστε κατάλληλο μήνυμα για το αν τελικά έγινε η εγκατάσταση. Έπειτα εκτυπώστε τη λίστα Η/Υ του εργαστηρίου.
* Ο 1ος Η/Υ του εργαστηρίου σταμάτησε να λειτουργεί, εκτελέστε την κατάλληλη ενέργεια με τη μέθοδο *pcStatus* και εκτυπώστε τη λίστα Η/Υ του εργαστηρίου.
* Ελέγξτε αν οι Η/Υ του εργαστηρίου μπορούν να εκτελέσουν μία εφαρμογή η οποία απαιτεί συχνότητα επεξεργαστή ≥ 2.8 GHz, RAM ≥ 12 GB και χώρο στο δίσκο 15.3 GB και τυπώστε κατάλληλο μήνυμα.

**Β. (4 μονάδες)** Να αναπτυχθεί ένα πρόγραμμα στο οποίο θα δημιουργήσετε ένα interface με το όνομα **Receipt** και μια συνάρτηση **float** **calculateAmount()**.

Έπειτα, να ορίσετε μία αφηρημένη κλάση **Purchase** η οποία αντιπροσωπεύει αγορές από ένα κατάστημα. Κάθε αγορά μπορεί να περιλαμβάνει μόνο ένα προϊόν. Η κλάση θα *υλοποιεί* το interface *Receipt* και θα έχει τα εξής πεδία και μεθόδους:

* **itemCode**: Κωδικός προϊόντος (int)
* **amount**: Κόστος αγορών (float)
* **Κατασκευαστές με ορίσματα και χωρίς**
* Συναρτήσεις **set** και **get** για όλα τα πεδία.
* **read()**: διαβάζει από το πληκτρολόγιο τα δεδομένα της κλάσης
* **String toString()**: επιστρέφει σε μορφή αλφαριθμητικού όλα τα πεδία της κλάσης
* **float calculateAmount()**: υπολογίζει και επιστρέφει το συνολικό κόστος της αγοράς με ΦΠΑ 24%.

Στη συνέχεια δημιουργήστε μια κλάση με το όνομα **OnlinePurchase** η οποία *κληρονομεί* την **Purchase** και έχει επιπρόσθετα τα εξής πεδία και μεθόδους:

* **address**: Διεύθυνση αποστολής (String)
* **shippingCost**: Κόστος μεταφορικών (float)
* **Κατασκευαστές με ορίσματα και χωρίς**
* Συναρτήσεις **set** και **get** για όλα τα πεδία.
* **read()**: διαβάζει από το πληκτρολόγιο τα δεδομένα της κλάσης
* **String toString()**: επιστρέφει σε μορφή αλφαριθμητικού όλα τα πεδία της κλάσης
* υλοποιεί τη συνάρτηση **float calculateAmount()**, υπολογίζει και επιστρέφει το συνολικό κόστος της αγοράς με ΦΠΑ 13%.

Στη συνέχεια δημιουργήστε μια κλάση με το όνομα **ShopPurchase** η οποία *κληρονομεί* την **Purchase** και έχει επιπρόσθετα τα εξής πεδία:

* **assistant**: Όνομα βοηθού (String)
* **tax**: Φόρος (int)
* **Κατασκευαστές με ορίσματα και χωρίς**
* Συναρτήσεις **set** και **get** για όλα τα πεδία.
* **String toString()**: επιστρέφει σε μορφή αλφαριθμητικού όλα τα πεδία της κλάσης
* υλοποιεί τη συνάρτηση **float calculateAmount()**, υπολογίζει και επιστρέφει το συνολικό κόστος της αγοράς με ΦΠΑ ίσο με τον φόρο που αντιστοιχεί στο προϊόν.

Τέλος, να αναπτυχθεί μια κλάση με το όνομα **Sales** η οποία θα έχει τα εξής πεδία και μεθόδους:

* **company**: Επωνυμία της εταιρείας (string)
* **day**: Ημερομηνία της μορφής ΗΗ/ΜΜ/ΕΕΕΕ (string)
* **Κατασκευαστής χωρίς ορίσματα**: αρχικοποιεί το όνομα σε κενό και το ArrayList (this.purchases = new ArrayList<>()).
* Συναρτήσεις **set** και **get**.
* **readSales()**: διαβάζει από το πληκτρολόγιο τα δεδομένα της κλάσης:
  + θα ζητά ημερομηνία
  + θα ρωτά το χρήστη τι θα είναι κάθε αντικείμενο (online αγορά ή αγορά στο κατάστημα) και θα εισάγει τα απαιτούμενα στοιχεία
  + θα αποθηκεύει τα αντικείμενα στο Arraylist
* **String toString()**: επιστρέφει σε μορφή αλφαριθμητικού όλα τα πεδία της κλάσης σε μορφή αποδείξεων
* **float calculateRevenue()**: υπολογίζει τα έσοδα της ημέρας αθροίζοντας τις εισπράξεις από online αγορές και αγορές στο κατάστημα και εκτυπώνει τις αντίστοιχες αποδείξεις με όλα τα στοιχεία τους.

**Γ. (2 μονάδες)** Σε συνέχεια της προηγούμενης άσκησης να ορίσετε μία κλάση **Test**, η οποία θα έχει τη συνάρτηση main, στην οποία θα κάνετε τα παρακάτω:

* Δηλώστε ένα αντικείμενο της κλάσης Sales με 4 αγορές από την κάθε κατηγορία.
* Τυπώστε τις αποδείξεις του και τα συνολικά έσοδα της ημέρας.
* Να δημιουργήσετε τις παρακάτω συναρτήσεις (όπου κρίνετε απαραίτητο):
  + βρίσκει την αγορά με τα μεγαλύτερα έσοδα και τυπώνει τα στοιχεία της.
  + συγκρίνει το κέρδος από τις online αγορές και τις αγορές στο κατάστημα.
  + βρίσκει τις πωλήσεις που έκανε ένα προϊόν και το συνολικό κέρδος και τα εκτυπώνει.