ΣΤΟΙΧΕΙΑ ΠΟΥ ΣΥΜΠΛΗΡΩΝΕΙ Ο ΦΟΙΤΗΤΗΣ

|  |  |
| --- | --- |
| Ονοματεπώνυμο φοιτητή | Ευάγγελος Μπάτσαλης |
| Αριθμός Μητρώου | ΑΜ119181 |
| Κωδικός Θ.Ε. | ΠΛΗ24 |
| Κωδικός Τμήματος | ΗΛΕ41 |
| Α/Α Γραπτής εργασίας | 2η ΕΡΓΑΣΙΑ |
| Ονοματεπώνυμο καθηγητή | ΦΙΤΣΙΛΗΣ ΠΑΝΑΓΙΩΤΗΣ |
| Σχόλια προς καθηγητή |  |

***Υπεύθυνη Δήλωση Φοιτητή****: Βεβαιώνω ότι είμαι συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία αυτής της εργασίας, είναι πλήρως αναγνωρισμένη και αναφέρεται, είτε στο σημείο «Σχόλια προς καθηγητή», είτε μέσα στην εργασία. Επίσης, έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς, είτε παραφρασμένες. Επίσης, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τη συγκεκριμένη Θεματική Ενότητα.*

|  |  |
| --- | --- |
| ΝΑΙ | Συμφωνώ και αποδέχομαι την ανωτέρω δήλωση |
|  | Δε συμφωνώ και δεν αποδέχομαι την ανωτέρω δήλωση (στην περίπτωση αυτή, ο Κ-Σ έχει δικαίωμα να μην αξιολογήσει την εργασία του φοιτητή) |

|  |
| --- |
| **Ημερομηνία ανακοίνωσης εργασίας:** Τετάρτη, 07/12/2021  **Ημερομηνία υποβολής εργασίας:** Τετάρτη, 19/01/2022, 11:59 μ.μ.  **Ημερομηνία υποβολής εργασίας με ατομική παράταση:** Τετάρτη 26/1/2022, 11:59 μ.μ.  *Ο κάθε φοιτητής δικαιούται μια ατομική παράταση σε όλη τη διάρκεια του ακαδημαϊκού έτους* |

ΣΤΟΙΧΕΙΑ ΠΟΥ ΣΥΜΠΛΗΡΩΝΕΙ Ο ΚΑΘΗΓΗΤΗΣ

|  |  |
| --- | --- |
| **Ημερομηνία αξιολόγησης** |  |
| **Τελικός βαθμός** |  |

**Σχόλια προς το φοιτητή**

**Αναλυτική αξιολόγηση**

|  |  |  |  |
| --- | --- | --- | --- |
| **Άσκηση** | **Περιγραφή** | **Ποσοστό %** | **Βαθμός** |
| 1.A | Δημιουργία Διαγράμματος Ακολουθίας | 10 |  |
| 1.B | Δημιουργία Διαγράμματος Κλάσεων | 5 |  |
| 2 | Χρήση Προτύπων Σχεδίασης | 10 |  |
| 3.Α | Κληρονομικότητα κλάσεων και υποσκέλιση μεθόδων | 15 |  |
| 3.Β | Υλοποίηση διαδικασιών | 25 |  |
| 3.Γ | Υλοποίηση προσομοίωσης με τα δεδομένα χρήστη | 10 |  |
| 3.Δ | Χειρισμός αλφαριθμητικών και εκτύπωση σε αρχείο κειμένου | 15 |  |
| 4 | Συμμόρφωση με τους κανόνες συγγραφής | 10 |  |
|  | Σύνολο | 100 |  |

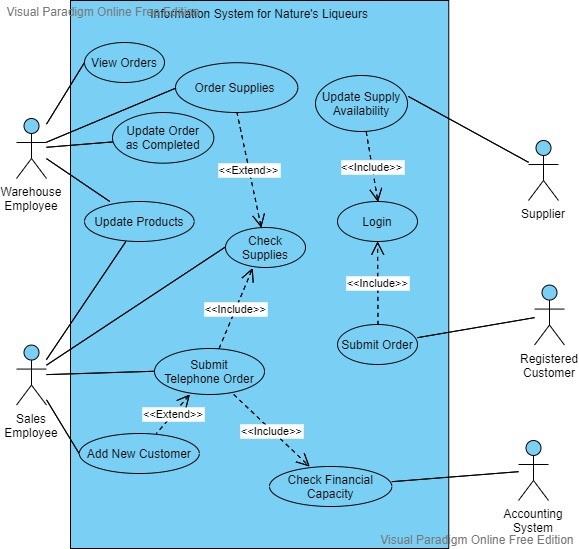
**Εκφώνηση 1ης Άσκησης**

**Πληροφοριακό Σύστημα Πωλήσεων και Προμηθειών της εταιρίας Nature’s Liqueurs.**

Το σύστημα χρησιμοποιείται από τη startup εταιρία Nature’s Liqueurs, η οποία παράγει βιολογικά λικέρ για διαβητικούς και τα πουλά σε εμπόρους (π.χ. κάβες, super market, καταστήματα βιολογικών προϊόντων). Η εταιρία χρησιμοποιεί το σύστημα για να καταγράφει τις παραγγελίες των πελατών της (τηλεφωνικές και ηλεκτρονικές), για να κάνει παραγγελίες στους προμηθευτές της και για να καταγράφει τα αποθέματά της. **Από την ανάλυση του πεδίου προβλήματος όπως έχει γίνει έως τώρα, έχει παραχθεί το διάγραμμα Περιπτώσεων Χρήσης (ΠΧ), καθώς και η λεκτική προδιαγραφή της ΠΧ, όπως δίνονται παρακάτω.**

**Διάγραμμα Περιπτώσεων Χρήσης**

**ΝΕΟ ΔΙΑΓΡΑΜΜΑ**



**Λεκτική Περιγραφή Περίπτωση Χρήσης «Καταχώρηση Τηλεφωνικής Παραγγελίας»**

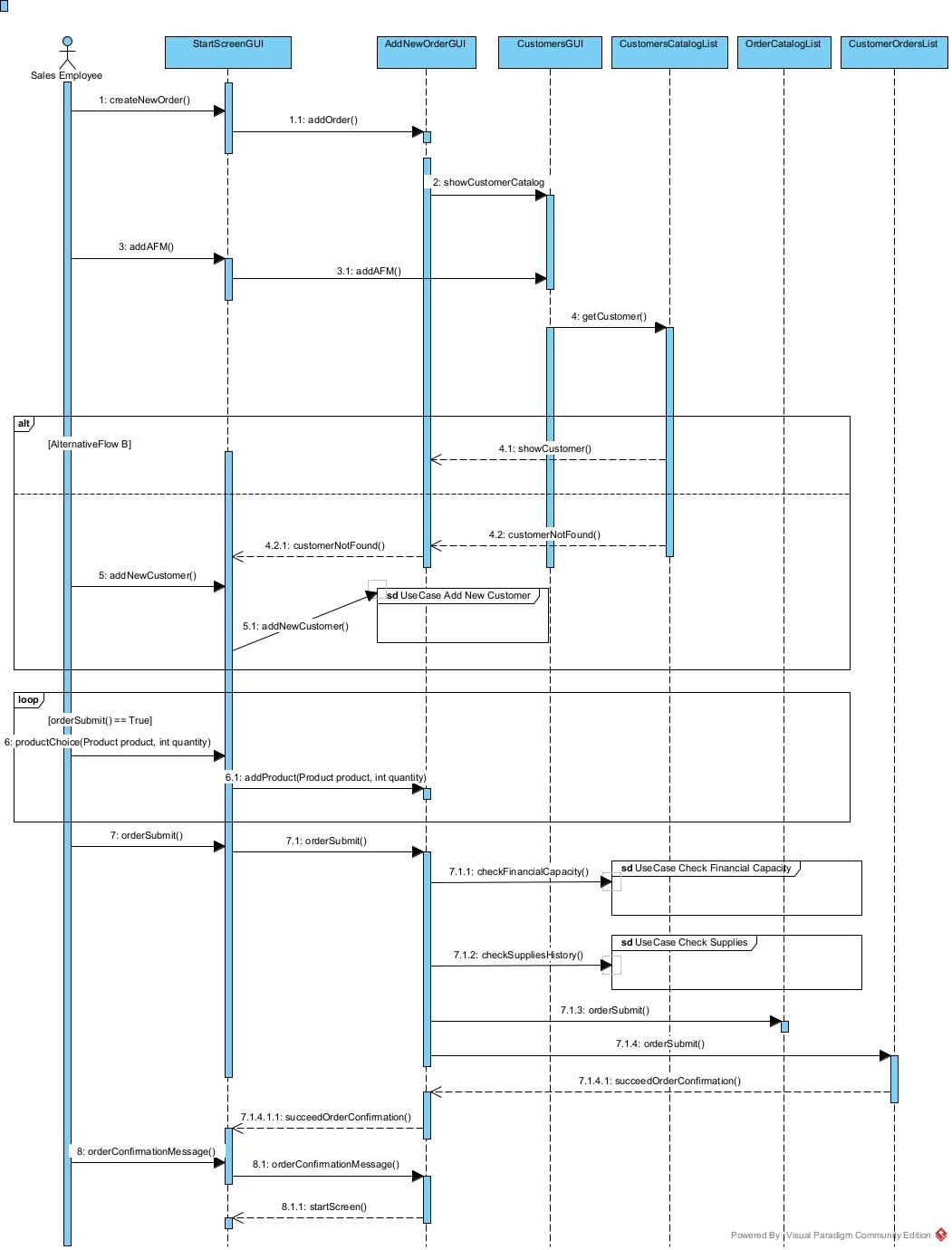
|  |  |  |  |
| --- | --- | --- | --- |
| **Όνομα ΠΧ** | | Καταχώρηση Τηλεφωνικής Παραγγελίας | |
| **Χειριστής** | | Υπάλληλος Πωλήσεων | |
| **Σύντομη Περιγραφή** | | | |
| Ο Υπάλληλος Πωλήσεων επιλέγει στο σύστημα τον πελάτη που έδωσε τηλεφωνική παραγγελία και καταχωρεί εκ μέρους του μια νέα παραγγελία βιολογικών λικέρ Nature’s Liqueurs | | | |
| **Ροή Γεγονότων** | | | |
| **Βασική Ροή** | | | |
| 1 | Ο Υπάλληλος Πωλήσεων επιλέγει στην Αρχική Οθόνη «Καταχώρηση Νέας Παραγγελίας» | | |
| 2 | Το σύστημα εμφανίζει την οθόνη επιλογής Πελάτη | | |
| 3 | Ο Υπάλληλος Πωλήσεων εισάγει το ΑΦΜ του Πελάτη | | |
| 4 | Το σύστημα εντοπίζει τον Πελάτη στον Κατάλογο Πελατών και εμφανίζει τα στοιχεία του στην οθόνη Καταχώρησης Παραγγελίας **[ΕΝΑΛΛΑΚΤΙΚΗ ΡΟΗ Α] [ΕΝΑΛΛΑΚΤΙΚΗ ΡΟΗ Β]** | | |
| 5 | Ο Υπάλληλος Πωλήσεων επιλέγει προϊόν και ποσότητα και πατάει το κουμπί “Προσθήκη” | | |
| 6 | Το σύστημα καταχωρεί στην Παραγγελία προϊόν και ποσότητα  (τα βήματα 5 και 6 επαναλαμβάνονται για όλα τα είδη της παραγγελίας) | | |
| 7 | Ο Υπάλληλος Πωλήσεων επιλέγει «Καταχώρηση Παραγγελίας» | | |
| 8 | Το σύστημα καλεί την ΠΧ «Check Financial Capacity» για να ελεγχθεί το ιστορικό συναλλαγών με το συγκεκριμένο Πελάτη | | |
| 9 | Το σύστημα καλεί την ΠΧ «Check Supplies» για να ελεγχθεί αν υπάρχουν τα απαιτούμενα αποθέματα για τη συγκεκριμένη Παραγγελία | | |
| 10 | Το σύστημα καταχωρεί την Παραγγελία στον Κατάλογο Παραγγελιών και στις παραγγελίες του Πελάτη που την υπέβαλε και εμφανίζει μήνυμα επιτυχούς καταχώρησης **[ΕΝΑΛΛΑΚΤΙΚΗ ΡΟΗ Γ]** **[ΕΝΑΛΛΑΚΤΙΚΗ ΡΟΗ Δ]** | | |
| 11 | Ο Υπάλληλος Πωλήσεων επιβεβαιώνει το μήνυμα | | |
| 12 | Το σύστημα εμφανίζει την Αρχική Οθόνη | | |
| **Εναλλακτικές Ροές** | | | |
| **Εναλλακτική Ροή Α**: ΔΕ ΒΡΕΘΗΚΕ ΤΟ ΑΦΜ – ΑΚΥΡΩΣΗ ΚΑΤΑΧΩΡΗΣΗΣ | | | |
| 1 | Το σύστημα δεν εντόπισε Πελάτη με το δοσμένο ΑΦΜ και εμφανίζει ενημερωτικό μήνυμα στον Υπάλληλο Πωλήσεων | | |
| 2 | Ο Υπάλληλος Πωλήσεων επιλέγει να ακυρώσει τη διαδικασία (επιλέγει «Ακύρωση») | | |
| 3 | Η εκτέλεση επιστρέφει στο βήμα 2 της βασικής ροής | | |
| **Εναλλακτική Ροή Β**: ΔΕΝ ΒΡΕΘΗΚΕ ΤΟ ΑΦΜ – ΚΑΤΑΧΩΡΗΣΗ ΝΕΟΥ ΠΕΛΑΤΗ | | | |
| 1 | Το σύστημα δεν εντόπισε Πελάτη με το δοσμένο ΑΦΜ και εμφανίζει ενημερωτικό μήνυμα στον Υπάλληλο Πωλήσεων | | |
| 2 | Ο Υπάλληλος Πωλήσεων επιλέγει να καταχωρήσει ένα νέο Πελάτη | | |
| 3 | Καλείται η ΠΧ «Add New Customer» | | |
| **Εναλλακτική Ροή Γ**: ΑΠΟΤΥΧΙΑ ΕΛΕΓΧΟΥ ΑΠΟΘΕΜΑΤΟΣ | | | |
| 1 | Το σύστημα διαπίστωσε ότι δεν υπάρχουν τα απαραίτητα αποθέματα για τα είδη της παραγγελίας και εμφανίζει σχετικό ενημερωτικό μήνυμα ακύρωσης της καταχώρησης παραγγελίας | | |
| 2 | Ο Υπάλληλος Πωλήσεων επιβεβαιώνει το μήνυμα | | |
| 3 | Το σύστημα επιστρέφει στην Αρχική Οθόνη | | |
| **Εναλλακτική Ροή Δ**: ΒΕΒΑΡΗΜΕΝΟ ΙΣΤΟΡΙΚΟ ΣΥΝΑΛΛΑΓΩΝ | | | |
| 1 | Το σύστημα μετά τον έλεγχο διαπιστώνει ότι ο συγκεκριμένος Πελάτης έχει βεβαρημένο ιστορικό προηγούμενων συναλλαγών και εμφανίζει μήνυμα ακύρωσης της καταχώρησης παραγγελίας | | |
| 2 | Ο Υπάλληλος Πωλήσεων επιβεβαιώνει το μήνυμα | | |
| 3 | Το σύστημα επιστρέφει στην Αρχική Οθόνη | | |
| **Προϋποθέσεις** | | | Καμία προϋπόθεση |
| **Μετασυνθήκες** | | | Χαρακτηρισμός μιας παραγγελίας ως «Under processing»  Κλήση της ΠΧ «Check Financial Capacity»  Κλήση της ΠΧ «Add New Customer»  Κλήση της ΠΧ «Έλεγχος Αποθέματος» |
| **Εξαιρέσεις (δεν συμπεριλαμβάνεται στο διάγραμμα ακολουθίας)** | | | - |

**Ερώτημα Α – Δημιουργία Διαγράμματος Ακολουθίας**

**Καλείστε να δημιουργήσετε το διάγραμμα ακολουθίας για την ΠΧ: «Καταχώρηση Τηλεφωνικής Παραγγελίας» απεικονίζοντας μόνο τη Βασική Ροή και την Εναλλακτική Ροή Β.** Από το διάγραμμα ΠΧ, παρακαλείστε να επικεντρωθείτε στις σχέσεις της ΠΧ «Καταχώρηση Τηλεφωνικής Παραγγελίας» (Submit Telephone Order).

|  |
| --- |
| ***Μαθησιακά Αποτελέσματα***  Στην άσκηση 1.A. θα σας δοθεί η δυνατότητα να κατανοήσετε την κατασκευή του «Διαγράμματος Ακολουθίας». Πιο συγκεκριμένα θα μπορέσετε:   * να περιγράψετε το Διάγραμμα Ακολουθίας, * να περιγράψετε τα μέλη που αυτό περιλαμβάνει, * να σχεδιάσετε τα σύμβολα που αυτό περιλαμβάνει, * να αντιστοιχίσετε τα μέλη με τα σύμβολα που αυτό περιλαμβάνει, * να συνδέσετε την λειτουργικότητα του κάθε συμβάντος με την έννοια της ιχνηλασιμότητας |

**Απάντηση**

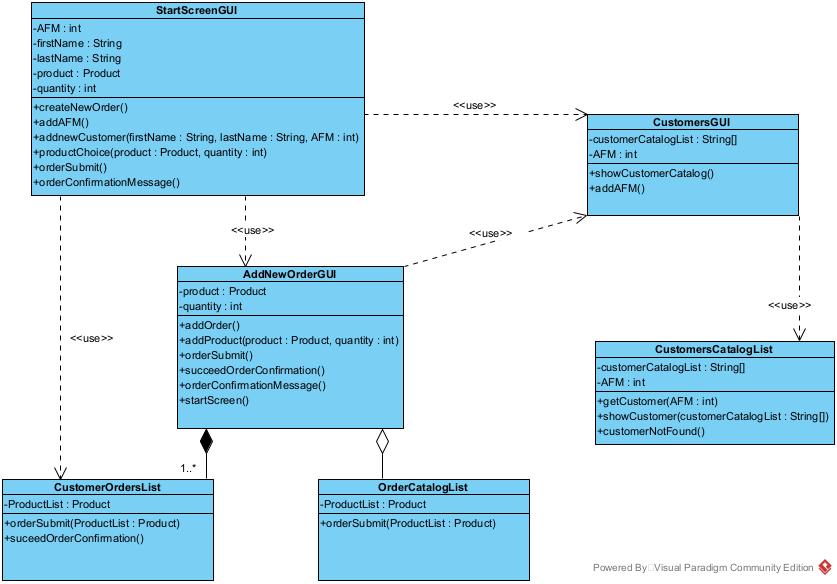


**Ερώτημα B – Δημιουργία Διαγράμματος Κλάσεων**

Με βάση το διάγραμμα ακολουθίας της άσκησης 1.Α, να σχεδιάσετε το λεπτομερές Διάγραμμα Κλάσεων. *Σημειώνουμε ότι το παραχθέν διάγραμμα κλάσεων θα πρέπει να είναι σε πλήρη συμφωνία με το διάγραμμα ακολουθίας*. Περιγράψτε τις κλάσεις του συστήματος, τις αρμοδιότητες τους (μέθοδοι και ιδιότητες), τις σχέσεις τους με άλλες κλάσεις και επεξηγήσετε γιατί επιλέξατε κάποια συγκεκριμένη σχέση. Όπου θεωρείτε ότι απαιτείται, να κάνετε τις αναγκαίες παραδοχές ώστε το διάγραμμα να είναι όσο το δυνατόν πλήρες.

|  |
| --- |
| ***Μαθησιακά Αποτελέσματα***  Στην άσκηση 1.B θα σας δοθεί η δυνατότητα να κατανοήσετε την κατασκευή του «Διαγράμματος Κλάσεων». Πιο συγκεκριμένα θα μπορέσετε:   * να παραθέσετε τα κύρια μέλη που αυτό περιλαμβάνει, * να παραθέσετε τα είδη των σχέσεων μεταξύ των κλάσεων, * να αντιστοιχίσετε το κάθε είδος σχέσης με το σύμβολο που την απεικονίζει, * να περιγράψετε τον τρόπο σύνταξης των ονομάτων των κλάσεων, * να περιγράψετε τον τρόπο σύνταξης των ιδιοτήτων των κλάσεων, * να περιγράψετε τον τρόπο σύνταξης των μεθόδων των κλάσεων και * να περιγράψετε με παράδειγμα το κάθε είδος σχέσης μεταξύ των κλάσεων. |

**Απάντηση:**



Όλες οι μέθοδοι έχουν δημιουργηθεί public ώστε να είναι προσβάσιμες από άλλες κλάσεις και οι ιδιότητες των αντικειμένων private.

**Κλάση StartScreenGUI:**

Βάση των παραδοχών της άσκησης 1 σχεδίασα μια κλάση την αρχική οθόνη (StartScreenGUI) στην οποία ο υπάλληλος πωλήσεων θα επιλέξει την καταχώρηση νέας παραγγελίας.

Με τις μεθόδους:

* createNewOrder() για την καταχώρηση παραγγελίας.
* addAFM() για την παραδοχή της άσκησης όπου ο υπάλληλος πωλήσεων εισάγει το αφμ του πελάτη.
* addNewCustomer(firstName, lastName, AFM) το οποίο παίρνει τα ορίσματα: Όνομα, Επώνυμο και ΑΦΜ για την επιλογή του εάν δεν βρεθεί ο πελάτης τότε βάση παραδοχής της άσκησης θα δημιουργηθεί νέος.
* productChoice(product : Product, quantity : int) λόγω παραδοχής της άσκησης όπου αναφέρει ότι ο πελάτης επιλέγει προιόν και ποσότητα. Γιαυτό τον λόγο δημιούργησα δύο ορίσματα το προιόν και την ποσότητα.
* orderSubmit() δίοτι αναφέρει η εκφώνηση της άσκησης: «Ο Υπάλληλος Πωλήσεων επιλέγει «Καταχώρηση Παραγγελίας».
* orderConfirmationMessage() για την παραδοχή της άσκησης η οποία αναφέρει ότι ο υπάλληλος πωλήσεων επιβεβαιώνει το μήνυμα.

Κατηγορήματα:

* AFM με την ιδιότητα αποθήκευσης ακεραίων αριθμών τύπου int.
* firstName με την ιδιότητα αποθήκευσης συμβολοσειρών τύπου String.
* lastName με την ιδιότητα αποθήκευσης συμβολοσειρών τύπου String.
* product με την ιδιότητα αποθήκευσης αντικειμένου τύπου Product.
* quantity με την ιδιότητα αποθήκευσης ακεραίων αριθμών τύπου int.

**Κλάση AddNewOrderGUI:**

Η δημιουργία της κλάσης αυτής έγινε για την εν λόγω παραδοχή της άσκησης η οποία αναφέρει την Καταχώρηση Νέας Παραγγελίας.

Με τις μεθόδους:

* addNeworder() για την προσθήκη νέας παραγγελίας.
* addProduct(product : Product, quantity : int) την προσθήκη ενός νέου προϊόντος στο οποίο παίρνει δύο ορίσματα το αντικείμενο Product και την ποσότητα quantity.
* orderSubmit(): για την παραδοχή όπου θα καταχωρηθεί η παραγγελία
* succeedOrderConfirmation(): Όπου θα είναι μια μέθοδος η οποία θα δείχνει ένα μήνυμα επιτυχούς ολοκλήρωσης καταχώρησης.
* orderConfirmationMessage() για την παραδοχή της άσκησης η οποία αναφέρει ότι ο υπάλληλος πωλήσεων επιβεβαιώνει το μήνυμα.

Με τα κατηγορήματα:

* product με την ιδιότητα αποθήκευσης αντικειμένου τύπου Product.
* quantity με την ιδιότητα αποθήκευσης ακεραίων αριθμών τύπου int.

**Κλάση CustomersGUI:**

Η κλάση αυτή δημιουργήθηκε με τον σκοπό της παραδοχής «Το σύστημα εμφανίζει την οθόνη επιλογής Πελάτη»

Με τις μεθόδους:

* showCustomerCatalog() είναι μια μέθοδος η οποία θα εμφανίζει τη λίστα του συγκεκριμένου πελάτη από τον κατάλογο πελατών.
* addAFM() για την παραδοχή της άσκησης όπου ο υπάλληλος πωλήσεων εισάγει το αφμ του πελάτη.

Με τα κατηγορήματα:

* customerCatalog : String[] όπου ο κατάλογος πελατών θα είναι μια λίστα τύπου String
* AFM με την ιδιότητα αποθήκευσης ακεραίων αριθμών τύπου int.

**Κλάση CustomersCatalogList:**

Η κλάση αυτή δημιουργήθηκε με τον σκοπό της παραδοχής «Το σύστημα εντοπίζει τον Πελάτη στον Κατάλογο Πελατών»

Με τις μεθόδους:

* getCustomer() όπου θα ζητείται να εμφανίζεται ο πελάτης βάση ορίσματος του ΑΦΜ.
* showCustomer() θα επιστρέφει τη λίστα με τον πελάτη οπό τον οποίο θα ζητείται από τη μέθοδο getCustomer().
* customerNotFound() θα επιστρέφει μέσω αυτής της μεθόδους την παραδοχή της άσκησης η οποία αναφέρει ότι δεν βρέθηκε ο πελάτης. Εναλλακτική ροής Β.

Με τα κατηγορήματα:

* customerCatalog : String[] όπου ο κατάλογος πελατών θα είναι μια λίστα τύπου String.
* AFM με την ιδιότητα αποθήκευσης ακεραίων αριθμών τύπου int.

**Κλάση OrderCatalogList:**

Η κλάση αυτή δημιουργήθηκε με τον σκοπό της παραδοχής «Το σύστημα καταχωρεί την Παραγγελία στον Κατάλογο Παραγγελιών». Χρησιμοποιήθηκε Aggregation διότι θεωρώ η λίστα παραγγελιών του πελάτη αποτελείται από την κλάση νέας παραγγελίας και οι παραγγελίες αυτές μπορούν να υπάρχουν ασχέτως αν μια παραγγελία ακυρωθεί.

Με τις μεθόδους:

* orderSubmit(ProductList) Όπου βάση παραδοχής θα γίνεται καταχώρηση της παραγγελίας ως όρισμα τη λίστα αντικειμένων των προιόντων.
* succeedOrderConfirmation(): Όπου θα είναι μια μέθοδος η οποία θα δείχνει ένα μήνυμα επιτυχούς ολοκλήρωσης καταχώρησης.

Κατηγορήματα:

* ProductList : Product[] η οποία είναι μια λίστα αντικειμένου προϊόντων.

**Κλάση CustomerOrdersList:**

Η κλάση αυτή δημιουργήθηκε με τον σκοπό της παραδοχής «Το σύστημα καταχωρεί την Παραγγελία στον Κατάλογο Παραγγελιών και στις παραγγελίες του Πελάτη». Χρησιμοποιήθηκε Composition διότι θεωρώ η λίστα παραγγελιών του συγκεκριμένου πελάτη αποτελείται από την κλάση νέας παραγγελίας και οι παραγγελίες αυτές δεν θα μπορούν να υπάρχουν ασχέτως ακυρωθεί το αντικείμενο πελάτης.

Με τις μεθόδους:

* orderSubmit(ProductList) Όπου βάση παραδοχής θα γίνεται καταχώρηση της παραγγελίας ως όρισμα τη λίστα αντικειμένων των προιόντων.

Με τα κατηγορήματα:

* ProductList : Product[] η οποία είναι μια λίστα αντικειμένου προϊόντων.

**Εκφώνηση 2ης Άσκησης**

Χρησιμοποιείστε το πρότυπο σχεδίασης (design pattern) Visitor και υλοποιήστε την παρακάτω απαίτηση μέσω διαγράμματος κλάσεων. Επεξηγείστε τις κλάσεις, τα πεδία, τις μεθόδους και τις συσχετίσεις μεταξύ των κλάσεων καθώς και τον τρόπο που ενσωματώσατε το πρότυπο στη σχεδίασή σας. Το πρότυπο σχεδίασης Visitor (δείτε πληροφορίες για το Visitor Design Pattern στο <https://refactoring.guru/design-patterns/visitor>) έχει σα στόχο την αναπαράσταση μιας λειτουργίας που πρόκειται να εφαρμοστεί στα στοιχεία μιας δομής αντικειμένων. Το πρότυπο επιτρέπει τον ορισμό μιας νέας λειτουργίας χωρίς την τροποποίηση των κλάσεων των στοιχείων στα οποία επιδρά.

***Απαίτηση***: Σχεδιάστε τα απαραίτητα αντικείμενα μιας εφαρμογής μισθοδοσίας σε ένα ιδιωτικό σχολείο όπου καταγράφεται η μηνιαία αμοιβή των εργαζομένων, το ποσό των επιδομάτων και οι μέρες αδειών του προσωπικού. Θεωρείστε πως το σχολείο έχει μόνο Διοικητικούς Υπαλλήλους και Εκπαιδευτικούς. Στους δεύτερους μόνο δίνεται η δυνατότητα να έχουν και υπερωριακή αμοιβή (overtime). Είναι λογικό πως ανάλογα με την εκάστοτε νομοθεσία είναι απαίτηση να τροποποιείται η μισθοδοτική πολιτική αλλά και οι άδειες σε τακτά χρονικά διαστήματα. Σχεδιάστε την απαίτηση αυτή αξιοποιώντας το πρότυπο σχεδίασης Visitor.

|  |
| --- |
| ***Μαθησιακά Αποτελέσματα***  Στην άσκηση 2 θα σας δοθεί η δυνατότητα να κατανοήσετε την εφαρμογή ενός προτύπου σχεδίασης. Πιο συγκεκριμένα θα μπορέσετε:   * να ταυτίσετε τους ρόλους ενός προτύπου σχεδίασης με κλάσεις ενός πραγματικού παραδείγματος, * να κατανοήσετε τη λειτουργία του προτύπου, * να περιγράψετε τον τρόπο με τον οποίο προσθέτουμε λειτουργικότητα σε ένα πρότυπο, * να κατανοήσετε τον τρόπο λειτουργίας της αρχής ανοιχτής-κλειστής σχεδίασης |

**Απάντηση:**

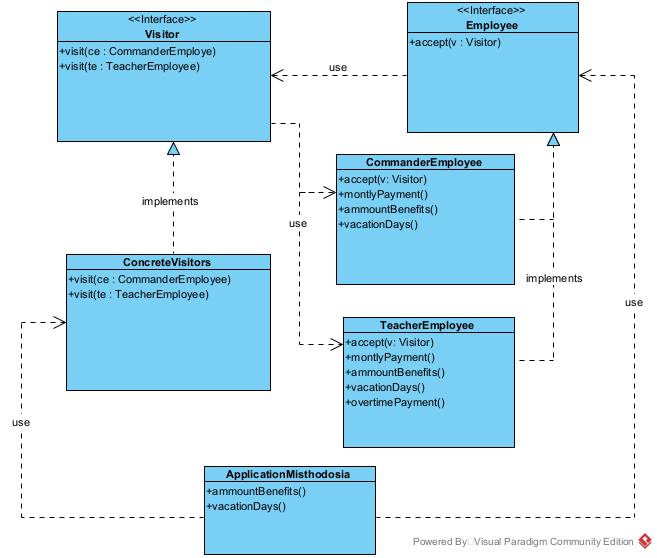
Όπως αναφέρει το πρότυπο σχεδίασης Visitor. Oι κλάσεις ElementA και ElementB αντικαταστάθηκαν από τις κλάσεις που αναφέρει η εκφώνηση για τους υπαλλήλους εκπαιδευτικούς και διοικητικούς. Οι οποίες έχουν χρήση από το interface Visitor όπου μέσω των μεθόδων visit παίρνουν ως ορίσματα τα αντικείμενα των κλάσεων CommanderEmployee και TeacherEmployee.

Όπου το interface Visitor υλοποιεί την κλάση interface Visitors με τη μέθοδο visit επισκέπτεται ουσιαστικά μέσω των αντικειμένων CommanderEmploye και TeacherEmployee.

H κλάση ApplicationMisthodosia κάνει χρήση την κλάση ConcreteVisitors όπως και την διεπαφή interface η οποία δέχεται και έχει σαν μέθοδο την accept η οποία παίρνει σαν όρισμα το αντικείμενο Visitor η οποία κάνει και χρήση την κλάση αυτή.

Η κλάση CommanderEmployee υλοποιεί το interface Employee και δέχεται σαν μέθοδοι φυσικά της διεπαφής με όρισμα το αντικείμενο Visitor και τις τρεις επιπλέον μεθόδους που αναφέρει η εκφώνηση montlyPayment() μηνιαία πληρωμή, ammountBenefits ποσό επιδομάτων, οι μέρες αδειών του προσωπικού vacationDays()

Η κλάση TeacherEmployee υλοποιεί το interface Employee και δέχεται σαν μέθοδοι φυσικά της διεπαφής με όρισμα το αντικείμενο Visitor και τις τρεις επιπλέον μεθόδους που αναφέρει η εκφώνηση montlyPayment() μηνιαία πληρωμή, ammountBenefits ποσό επιδομάτων, οι μέρες αδειών του προσωπικού vacationDays() τέλος έχει τη μέθοδο overtimePayment() όπου είναι η υπερωριακή αμοιβή.

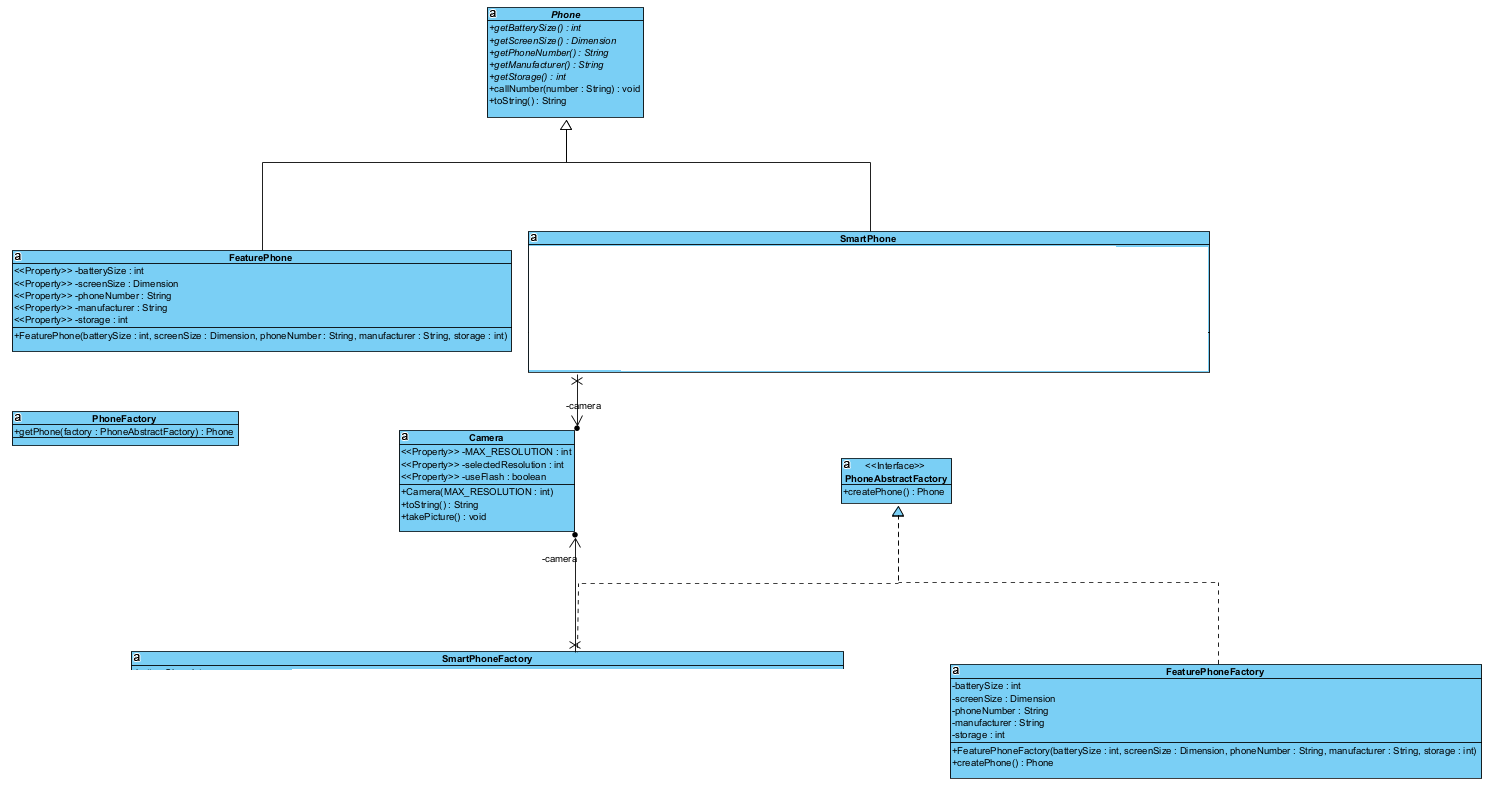


**Εκφώνηση 3ης Άσκησης**

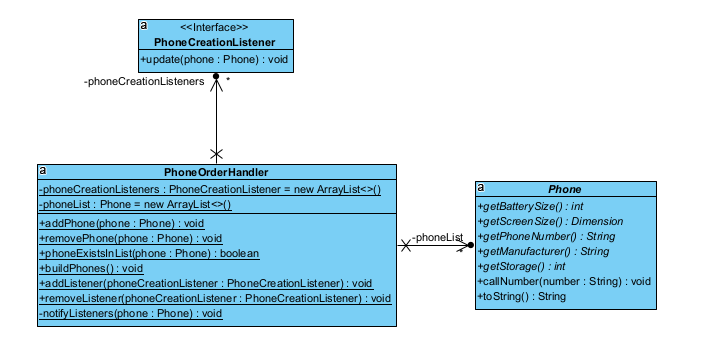
Σας δίνεται το ημιτελές Java Project “plh24ge2.zip”, το οποίο διαθέτει κλάσεις και διεπαφές χωρισμένα σε 3 πακέτα, σχετικά με ένα απλοποιημένο σύστημα προσομοίωσης γραμμής παραγωγής κινητών τηλεφώνων και πελατών.

Χρησιμοποιήστε τις δοθείσες κλάσεις / διεπαφές και συμπληρώστε ό,τι κρίνετε απαραίτητο (νέες κλάσεις / διεπαφές, πεδία, μεθόδους) ώστε να υλοποιήσετε το εν λόγω Project, απαντώντας στα ερωτήματα που ακολουθούν.

Περιγραφή: Στο εν λόγω project υλοποιούνται δύο πρότυπα σχεδίασης, συγκεκριμένα το “Abstract Factory” (<https://refactoring.guru/design-patterns/abstract-factory>) και το “Observer” (<https://refactoring.guru/design-patterns/observer>). Το πρότυπο Abstract Factory χρησιμοποιείται, τροποποιημένο, για την παραγωγή κινητών τηλεφώνων οποιουδήποτε τύπου, όπου στο συγκεκριμένο project εξειδικευόμαστε σε Featurephones και Smartphones (και γενικά μπορεί να χρησιμοποιηθεί σε έναν μεγάλο αριθμό αντίστοιχων προβλημάτων/προδιαγραφών, όπου από γενικούς τύπους θέλουμε να φτάσουμε στην κατασκευή ειδικότερων τύπων). Η εν λόγω χρήση του προτύπου στο project αναπαρίσταται με τη χρήση διαγράμματος κλάσεων στην επόμενη εικόνα (πακέτο eap.abstractfactory). Τα περιεχόμενα των κλάσεων SmartPhone και SmartPhoneFactory δεν φαίνονται στην εικόνα λόγω του ότι αποτελούν ζητούμενα.



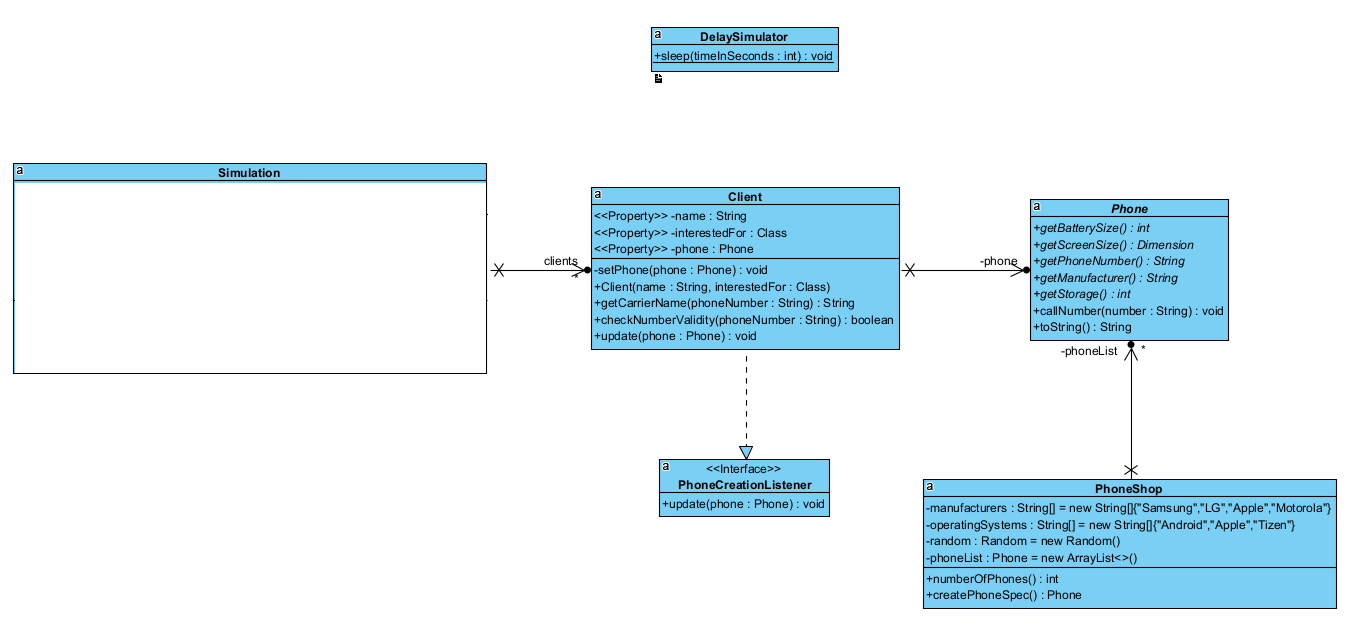
Το πρότυπο Observer χρησιμοποιείται, τροποποιημένο, για την «εγγραφή» των πελατών ως «ενδιαφερόμενους» για την απόκτηση κινητών τηλεφώνων, με αποτέλεσμα την άμεση ενημέρωσή τους όποτε προκύψει ένα τηλέφωνο. Η εν λόγω χρήση του προτύπου στο project αναπαρίσταται με τη χρήση διαγράμματος κλάσεων στην επόμενη εικόνα (πακέτο eap.observer).



Και για τα 2 παραπάνω πρότυπα που αναφέρονται δεν χρειάζεται να υλοποιήσετε νέα αρχεία, είναι ολοκληρωμένα σχεδιαστικά, σχετικά με τα πρότυπα που εξυπηρετούν. Θα χρειαστεί να προσθέσετε κώδικα, όπου ζητείται εντός των κλάσεων που τα χρησιμοποιούν, σύμφωνα με τα ερωτήματα της άσκησης.

Λογική: Η λογική εκτέλεσης του εν λόγω project έχει ως εξής: Το πρόγραμμα (εκκίνηση του προγράμματος από τη μέθοδο main της κλάσης Simulation, πακέτο eap.simulate) αρχικά ζητά από τον χρήστη να δηλώσει τον αριθμό των κινητών τηλεφώνων που πρόκειται να δοθούν στη συνέχεια στη γραμμή παραγωγής, καθώς και τον αριθμό των πελατών που θα δημιουργηθούν και θα αναμείνουν για την ενδεχόμενη παραγωγή των τηλεφώνων που τους ενδιαφέρουν. Οι πελάτες, με τυχαίο τρόπο, ενδιαφέρονται είτε για Featurephone, είτε για Smartphone (ένα ο καθένας). Εν συνεχεία, δημιουργείται ένα αντικείμενο της κλάσης “PhoneShop”, το οποίο είναι ουσιαστικά υπεύθυνο για τη δημιουργία των προδιαγραφών, οι οποίες πρέπει να είναι τυχαίες, των κινητών που θέλουμε να παραχθούν. Οι εν λόγω προδιαγραφές τηλεφώνων μεταφέρονται στην κλάση “PhoneOrderHandler”, η οποία πρόκειται να εκτελέσει τις κατασκευές αυτών, καθώς και να ενημερώσει τους ενδιαφερόμενους πελάτες (χρήση της μεθόδου “buildPhones”). Ο κάθε πελάτης ο οποίος έχει τελικά λάβει ένα κινητό τηλέφωνο, το χρησιμοποιεί, στην περίπτωση Featurephone κάνοντας μια κλήση σε έναν αριθμό, ενώ στην περίπτωση του Smartphone τραβώντας μια φωτογραφία. Το πρόγραμμα ολοκληρώνεται αναγράφοντας τον αριθμό των πελατών που, ενδεχομένως, έμειναν χωρίς να πάρουν ένα κινητό τηλέφωνο.

Ακολουθεί και το διάγραμμα κλάσεων του πακέτου eap.simulate, για λόγους πληρότητας. Τα περιεχόμενα της κλάσης Simulate δεν φαίνονται στην εικόνα λόγω του ότι αποτελούν ζητούμενα.



Σημειώνεται ότι τα πεδία εντός των κλάσεων (χαρακτηριστικά και λειτουργίες) στα δοθέντα διαγράμματα κλάσεων ενδεχομένως να είναι διαφορετικά στην τελική, δική σας υλοποίηση, χωρίς αυτό να είναι λάθος εφόσον υλοποιήσετε σωστά τα δεδομένα που σας έχουν δοθεί από τις εκφωνήσεις των ερωτημάτων.

Στη συνέχεια ακολουθεί ένας αριθμός από απαιτήσεις και παραδοχές που αφορούν στο συγκεκριμένο project:

1. Η κλάση “Simulation” περιέχει τη μέθοδο main, από την οποία ξεκινάει την εκτέλεσή του το πρόγραμμα και κάνει την προσομοίωση του project:
   1. Διαθέτει έναν πίνακα (names) με τα ονόματα χρηστών (όσων) που θα χρησιμοποιηθούν (μπορείτε να αφήσετε τον πίνακα ως έχει)
   2. Ζητά από τον χρήση να εισάγει τον αριθμό των τηλεφώνων που πρόκειται να δοθούν ως παραγγελίες για να φτιαχτούν
   3. Ζητά από τον χρήστη να εισάγει τον αριθμό των πελατών που θα περιμένουν για να πάρουν ένα κινητό (μέγιστος αριθμός 10)
   4. Στη συνέχεια προσομοιώνει το κατάστημα PhoneShop, καθώς και τους πελάτες, επιλέγοντας ένα τυχαίο όνομα για τον κάθε πελάτη. Επιπλέον, όπως προαναφέρθηκε, σε κάθε πελάτη θα πρέπει να αντιστοιχηθεί ένα είδος τηλεφώνου (FeaturePhone ή SmartPhone) με τυχαίο τρόπο. Κάθε ένας πελάτης (άρα κάθε όνομα) μπορεί να λάβει ένα τηλέφωνο
   5. Τέλος εκτυπώνει ως αναφορά το πόσοι πελάτες δεν έλαβαν κάποιο τηλέφωνο (σύμφωνα με τα χαρακτηριστικά που ζήτησαν, ή αν ήταν περισσότεροι από τα τηλέφωνα που παραγγέλθηκαν)
2. Η γραμμή παραγωγής κινητών τηλεφώνων υποστηρίζει στην τρέχουσα υλοποίηση τα Featurephones (παλαιότερου τύπου κινητά τηλέφωνα, χωρίς λειτουργικό και κάμερα) και τα Smartphones (σύγχρονα κινητά τηλέφωνα, με λειτουργικό σύστημα και κάμερα). Συγκεκριμένα, τα Smartphones έχουν όλες τις δυνατότητες των Featurephones, ενώ επιπλέον διαθέτουν λειτουργικό σύστημα και κάμερα.
3. Τα αρχεία κλάσεων/διεπαφών που αφορούν στο πρότυπο Abstract Factory βρίσκονται στο πακέτο “eap.abstractfactory”, τα αρχεία που αφορούν στο πρότυπο Observer βρίσκονται στο πακέτο “eap.observer”, ενώ τα υπόλοιπα αρχεία, που αφορούν στις προσομοιώσεις, στο πακέτο “ eap.simulate”.
4. Στο project, όπου είναι δυνατόν, χρησιμοποιούνται private fields σε συνδυασμό με getters/setters (όπου χρειάζονται), για τον καλύτερο και ασφαλέστερο χειρισμό των δεδομένων.
5. Η κλάση “DelaySimulator” διαθέτει τη στατική μέθοδο sleep(int timeInSeconds) με την οποία προσομοιώνει μια χρονική καθυστέρηση για τον δοθέντα χρόνο σε δευτερόλεπτα, χωρίς τη χρήση νημάτων (threads).
6. Η διεπαφή “PhoneCreationListener” χρησιμοποιείται αφενός στην κλάση “PhoneOrderHandler”, όσο και σε όποια άλλη κλάση θέλει να δηλώσει «ενδιαφέρον παρακολούθησης/παρατήρησης» πάνω σε συγκεκριμένα αντικείμενα. Ως εκ τούτου, οι ενδιαφερόμενοι για κινητά τηλέφωνα Clients, υλοποιούν το εν λόγω Interface, ώστε στη συνέχεια να μπορούν να ενημερώνονται, μέσω της κλάσης “PhoneOrderHandler”, για όποτε αυτό συμβεί.
7. Η κλάση “PhoneOrderHandler” είναι υπεύθυνη για τον χειρισμό των παραγγελιών ως προς την υλοποίηση των παραγγελιών, καθώς και για την ενημέρωση των ενδιαφερόμενων πελατών. Εντός της κλάσης, η δημιουργία ενός νέου τηλεφώνου γίνεται με την κλήση της μεθόδου “PhoneFactory.getPhone”. Επιπλέον, διαθέτει τη μέθοδο “printCreatedPhoneSpecsToTextFile”, η οποία σώζει τα βασικά χαρακτηριστικά των τηλεφώνων που παράχθηκαν σε αρχείο κειμένου.
8. Η κλάση “PhoneShop” είναι υπεύθυνη ώστε οι προδιαγραφές των κινητών τηλεφώνων (όσα πεδία διαθέτουν) να παράγονται με τυχαίο τρόπο. Επιπλέον, διατηρεί και μια λίστα στην οποία καταγράφονται τα τηλέφωνα για τα οποία παρήχθησαν προδιαγραφές (και πρόκειται να κατασκευαστούν).
9. Η κλάση “Client” που αφορά στον πελάτη, αναλαμβάνει μεταξύ άλλων, όποτε ειδοποιείται για την παραγωγή νέου κινητού τηλεφώνου να κάνει τα εξής:
   1. Να ελέγχει αν το κινητό που παράχθηκε είναι τέτοιου τύπου που να ενδιαφέρει τον πελάτη (Featurephone ή Smartphone). Έχοντας στη διάθεσή του το τηλέφωνο που τον ενδιαφέρει, στη περίπτωση που είναι Featurephone πραγματοποιεί μια κλήση σε ένα τηλέφωνο (δεν χρειάζεται να είναι τυχαίο), ενώ στην περίπτωση που διαθέτει Smartphone τραβάει μια φωτογραφία σε ανάλυση και χρήση φλας της επιλογής του (δεν χρειάζεται να είναι τυχαία) (μέθοδος usePhone)
   2. Εφόσον ο πελάτης βρίσκει το τηλέφωνο που τον ενδιαφέρει, πρέπει να αφαιρεί το τηλέφωνο από το σύνολο των διαθέσιμων τηλεφώνων (για να μην το πάρει κάποιος άλλος), να απεγγράφεται από τη λίστα των ενδιαφερόμενων πελατών (ώστε να μην ενημερωθεί ξανά), καθώς και να αφαιρείται από το σύνολο των πελατών που αναμένουν για κινητό (μέθοδος update)
   3. Να ελέγχει και να εκτυπώνει κατά πόσο ο τηλεφωνικός αριθμός του τηλεφώνου που έλαβε είναι έγκυρος (οι πληροφορίες βρίσκονται στην περιγραφή της μεθόδου “checkNumberValidity”)
   4. Να ελέγχει και να εκτυπώνει το όνομα του παρόχου κινητής τηλεφωνίας για τον τηλεφωνικό αριθμό που έλαβε (πληροφορίες στις περιγραφές του πίνακα «carriers» και της μεθόδου ”getCarrierName” αντίστοιχα

Ενδεικτική εκτέλεση του προγράμματος (ενδεικτικά δοκιμαστικά δεδομένα):

Please enter number of phones to be ordered:3

Please enter number of clients waiting for new phones (MAX:10):2

3 phone orders have been placed by the PhoneShop!

2 clients are waiting to buy a new phone!

##########-Phone Build Process Started-##########

$$$$-New Phone-$$$$

Battery(mAh): 4400

Screen: java.awt.Dimension[width=2928,height=1581]

Number: +306981066085

Manufacturer: Apple

Storage(GB): 144

Camera: 72 MP

Operating System: Tizen

Has been created

$$$$-----------$$$$

$$$$-New Phone-$$$$

Battery(mAh): 1445

Screen: java.awt.Dimension[width=1951,height=2214]

Number: +306998519113

Manufacturer: Samsung

Storage(GB): 12

Camera: 90 MP

Operating System: Android

Has been created

$$$$-----------$$$$

$$$$-New Phone-$$$$

Battery(mAh): 3067

Screen: java.awt.Dimension[width=1032,height=2126]

Number: +306981911515

Manufacturer: Motorola

Storage(GB): 5

Has been created

$$$$-----------$$$$

Hi, I am Eleni(FeaturePhone) and I got my new phone!

Phone specs:

Battery(mAh): 3067

Screen: java.awt.Dimension[width=1032,height=2126]

Number: +306981911515

Manufacturer: Motorola

Storage(GB): 5

Phone number valid:true

Phone number carrier:Cosmote

------------------------

Phone call from +306981911515 to +30123456789

Call in progress...

Call terminated.

------------------------

########-Phone Specs saved to output.txt-########

###########-Phone Build Process Ended-###########

%%%%%----Report----%%%%%

1 clients did not get a phone...

%%%%%--------------%%%%%

Ενδεικτικό αρχείο output.txt, για την παραπάνω εκτέλεση (από τη μέθοδο printCreatedPhoneSpecsToTextFile):

1.

Battery(mAh): 4400

Screen: java.awt.Dimension[width=2928,height=1581]

Number: +306981066085

Manufacturer: Apple

Storage(GB): 144

Camera: 72 MP

Operating System: Tizen

2.

Battery(mAh): 1445

Screen: java.awt.Dimension[width=1951,height=2214]

Number: +306998519113

Manufacturer: Samsung

Storage(GB): 12

Camera: 90 MP

Operating System: Android

3.

Battery(mAh): 3067

Screen: java.awt.Dimension[width=1032,height=2126]

Number: +306981911515

Manufacturer: Motorola

Storage(GB): 5

**Ερώτημα Α – Κληρονομικότητα κλάσεων και υποσκέλιση (override) μεθόδων**

Με βάση τις απαιτήσεις/προδιαγραφές, να υλοποιήσετε τον κώδικα που λείπει για τις κλάσεις SmartPhone και SmartPhoneFactory, λαμβάνοντας υπόψη και τα όποια σχόλια υπάρχουν εντός του κώδικα που σας δίνεται.

|  |
| --- |
| ***Μαθησιακά Αποτελέσματα***  Στην άσκηση 3.Α. θα μάθετε:   * να ορίσετε με ακρίβεια την έννοια της κλάσης, * να κατανοήσετε τα βασικά δομικά στοιχεία των κλάσεων και των διεπαφών * να χρησιμοποιήσετε κλάση η οποία υλοποιεί μια διεπαφή * να κατανοήσετε την έννοια της μεθόδου * να εξηγήσετε τις βασικές διαφορές ενός κατασκευαστή από μια απλή μέθοδο, * να υλοποιήσετε κληρονομικότητα και υποσκέλιση |

**Απάντηση**

*Εισάγετε τον κώδικα των δύο κλάσεων Java που φτιάξατε. Φροντίστε ο κώδικας να είναι μορφοποιημένος κατάλληλα και να είναι ευανάγνωστος. Θα πρέπει να* ***υπάρχει τεκμηρίωση με μορφή σχολίων****.*

*Εάν δεν έχετε δώσει απάντηση γράψτε με κεφαλαία γράμματα, ΔΕΝ ΑΠΑΝΤΗΘΗΚΕ.*

*Εάν εν γνώση σας δίνετε ελλιπή απάντηση γράψτε με κεφαλαία γράμματα, ΕΛΛΙΠΗΣ ΑΠΑΝΤΗΣΗ.*

**//ΚΛΑΣΗ SmartPhone**

package eap.abstractfactory;

//εισαγωγή βιβλιοθήκης Dimension όπως και αναγράφεται στο manual της oracle ώστε να λειτουργήσει το κατηγόρημα screenSize του τελεστή Dimension.

import java.awt.Dimension;

/\*Η κλάση αυτή επεκτείνει την κλάση Phone.

Διαθέτει επιπλέον τα χαρακτηριστικά camera (της δοθείσας αντίστοιχης κλάσης) και το αλφαριθμητικό operatingSystem

Θα πρέπει εντός της υλοποίησής της να υποσκελίσετε (να κάνετε override) τη μέθοδο toString η οποία έχει κληρονομηθεί από την κλάση Phone.

Ωστόσο, στη νέα υλοποίηση της μεθόδου toString Θα πρέπει ο κώδικάς σας να χρησιμοποιεί την κληρονομηθείσα μέθοδο και να συμπληρώνει το

επιστρεφόμενο αποτέλεσμα με τα νέα πεδία που εισήχθησαν στην κλάση SmartPhone.\*/

//χρησιμοποίησα extends διότι η αρχική κλάση όπου και κληρονομεί τις μεθόδους είναι η phone και είναι κλάση abstract.

public class SmartPhone extends Phone{

//Εισαγωγή των χαρακτηριστικών της κλάσης Phone η οποία είναι abstract

private final int batterySize;

private final Dimension screenSize;

private final String phoneNumber;

private final String manufacturer;

private final int storage;

//επλιπλέον χαρακτηριστικά camera καιτο αλφαριθμητικό operatingSystem

private final Camera camera;

//επειδή το χαρακτηριστικό Camera αντικείμενο Θα το δημιουργήσω ώστε να δεσμευτεί ο κατάλληλος χώρος στη μνήμη.

private final String operatingSystem;

//Δημιουργία Constructor της κλάσης smartphone συμπεριλαμβάνοντας τα επιπλέον χαρακτηριστικά της κλάσης

public SmartPhone(int batterySize, Dimension screenSize, String phoneNumber, String manufacturer, int storage, Camera camera, String operatingSystem){

this.batterySize = batterySize;

this.screenSize = screenSize;

this.phoneNumber = phoneNumber;

this.manufacturer = manufacturer;

this.storage = storage;

this.camera = camera;

this.operatingSystem = operatingSystem;

}

//Δημιουργία των μεθόδων της phone abstract method

//Δημιουργία Getters

@Override

public int getBatterySize(){

return batterySize;

}

@Override

public Dimension getScreenSize(){

return screenSize;

}

@Override

public String getPhoneNumber(){

return phoneNumber;

}

@Override

public String getManufacturer(){

return manufacturer;

}

@Override

public int getStorage(){

return storage;

}

public Camera getCamera(){

return camera;

}

public String getOperatingSystem(){

return operatingSystem;

}

//Δεν θα γίνει η δημιουργία των setters διότι είναι final και δεν μπορούν να παραμετροποιηθούν και οι μεταβλητές θα περάσουν από την κλάση asbstacτ

//μέθοδος toString μέθοδος overide από της κλάσης που κληρονόμησε

@Override

public String toString() {

/\*λόγω του ότι η εκφώνηση αναφέρει "ο κώδικάς σας να χρησιμοποιεί την κληρονομηθείσα μέθοδο" θα χρησιμοποιήσω τον τελεστή super για να αποκτήσω

πρόσβαση στη μέθοδο της γονικής κλάσης\*/

return super.toString()

//προσθέτω τα επιπλέον χαρακτηριστικά camera και operating system όπως αναφέρει η εκφώνηση

+"\n"

+"Camera: "+getCamera()+"MP"+"\n"

+"Operatin System: "+getOperatingSystem();

}//end of toString method

}//end of class

**//ΚΛΑΣΗ SmartPhoneFactory**

package eap.abstractfactory;

//εισαγωγή βιβλιοθήκης java.awt.\*; όπως και αναγράφεται στο manual της oracle ώστε να λειτουργήσει το κατηγόρημα screenSize του τελεστή Dimension.

import java.awt.\*;

//Η κλάση αυτή υλοποιεί τη διεπαφή PhoneAbstractFactory

//δημιουργία αντικειμένων όπου και θα χρειαστεί να στείλουμε στη μέθοδο createPhone() της κλάσης που κληρονομεί από το interface PhoneAbstractFactory

public class SmartPhoneFactory implements PhoneAbstractFactory{

private final int batterySize;

private final Dimension screenSize;

private final String phoneNumber;

private final String manufacturer;

private final int storage;

private final Camera camera;

private final String operatingSystem;

//Δημιουργία constructor με τα χαρακτηριστικά της κλάσης συμπεριλαμβανόμενου και τα επιπλέον αντικείμενα camera και operating System

public SmartPhoneFactory(int batterySize, Dimension screenSize, String phoneNumber, String manufacturer, int storage, Camera camera, String operatingSystem){

this.batterySize = batterySize;

this.screenSize = screenSize;

this.phoneNumber = phoneNumber;

this.manufacturer = manufacturer;

this.storage = storage;

this.camera = camera;

this.operatingSystem = operatingSystem;

}

/\*γίνεται override και κληρορνομεί από την κλάση PhoneAbstractFactory μια μέθοδο createPhone() ενός αντικειμένου Phone με τα χαρακτηριστικά ενός SmartPhone

συμπεριλαμβανόμενου και τα επιπλέον αντικείμενα camera και operating system

όπου και τα στέλνουμε στον constructor της συγκεκριμένης κλάσης Phone\*/

@Override

public Phone createPhone(){

return new SmartPhone(batterySize,screenSize,phoneNumber,manufacturer,storage,camera,operatingSystem);

}

}//end of Class

**Ερώτημα Β – Υλοποίηση διαδικασιών**

Με βάση τις απαιτήσεις/προδιαγραφές, τον κώδικα που έχετε λάβει, καθώς και τα σχόλια εντός των αρχείων κώδικα Java, να υλοποιήστε τις παρακάτω μεθόδους:

* 1. Στην κλάση **Client** τη μέθοδο “**update”** (μελετήστε την απαίτηση/παραδοχή 9.b)μέσα στην οποία θα καλέσετεκαι τις μεθόδους **“getCarrierName”** και **“checkNumberValidity”,** οι υλοποιήσεις των οποίων ζητούνται στο ερώτημα Δ.
  2. Στην κλάση **PhoneShop** τη μέθοδο “**createPhoneSpec”,** η οποία παράγει τα χαρακτηριστικά των τηλεφώνων (ανάλογα με το είδος του τηλεφώνου) με τυχαίο τρόπο για κάθε τηλέφωνο και ενημερώνει και τη λίστα τηλεφώνων με τις προδιαγραφές τους (λίστα phoneList).

|  |
| --- |
| ***Μαθησιακά Αποτελέσματα***  Στην άσκηση 3.Β. θα μάθετε:   * Να υλοποιείτε μεθόδους ως υλοποίηση διεπαφών (Interfaces) * Να προσθαφαιρείτε στοιχεία από λίστες αντικειμένων * Να παράγετε αντικείμενα με τυχαία χαρακτηριστικά * Να κατανοείτε τη λογική πίσω από πρότυπα σχεδίασης της Java |

**Απάντηση**

*Εισάγετε τον κώδικα των δύο μεθόδων Java που φτιάξατε. Φροντίστε ο κώδικας να είναι μορφοποιημένος κατάλληλα και να είναι ευανάγνωστος. Θα πρέπει να* ***υπάρχει τεκμηρίωση με μορφή σχολίων****.*

*Εάν δεν έχετε δώσει απάντηση γράψτε με κεφαλαία γράμματα, ΔΕΝ ΑΠΑΝΤΗΘΗΚΕ.*

*Εάν εν γνώση σας δίνετε ελλιπή απάντηση γράψτε με κεφαλαία γράμματα, ΕΛΛΙΠΗΣ ΑΠΑΝΤΗΣΗ.*

**//Μέθοδος Update της κλάσης Client**

/\*

(μέθοδος update)

Εφόσον ο πελάτης βρίσκει το τηλέφωνο που τον ενδιαφέρει,

πρέπει να αφαιρεί το τηλέφωνο από το σύνολο των διαθέσιμων τηλεφώνων (για να μην το πάρει κάποιος άλλος),

να απεγγράφεται από τη λίστα των ενδιαφερόμενων πελατών (ώστε να μην ενημερωθεί ξανά),

καθώς και να αφαιρείται από το σύνολο των πελατών που αναμένουν για κινητό

\*/

@Override

public void update(Phone phone) {

//βάση παραδοχή της εργασίας 9α γίνεται έλεγχος τύπου του τηλεφώνου που ενδιαφερει τον πελάτη

String temp = null;

if (phone instanceof FeaturePhone) {

temp = "FeaturePhone";

} else if (phone instanceof SmartPhone) {

temp = "SmartPhone";

}

PhoneCreationListener phoneCreationListener = null;

//χρησιμοποίησα to isInstance() https://www.geeksforgeeks.org/instanceof-operator-vs-isinstance-method-in-java/ για να ελέγξω τπ phone να είναι αντικείμενο της κλάσης insterestedFor

//χρησιμοποίησα (PhoneOrderHandler.phoneExistsInList(phone) ώστε να ελέγεται το τηλέφωνο μιαφορά και να μην το ξανααίρνει ο πελάτης ο ίδιος δύο φορές

if ((this.interestedFor.isInstance(phone)) && (PhoneOrderHandler.phoneExistsInList(phone))){

String carrierName = this.getCarrierName(phone.getPhoneNumber());

System.out.println("hi i am " + this.name + "(" + temp + ") and I got my new phone!\n"

+ "Phone specs: \n"

+ phone.toString() + "\n"

+ //από το αντικείμενο phone καλώ τη μέθοδο toString ώστε να τυπώσει στην έξοδο της κοσνόλας τα χαρακτηριστικά τηλεφώνου

"Phone number valid: " + this.checkNumberValidity(phone.getPhoneNumber()) + "\n"

+ //καλώ τη μέθοδο checkNumberValidity αυτή της κλάσης( με τον τελεστή this) βάζοντας ως παράμετρο το getter getPhoneNumber() του αντικειμένου phone ώστε να στείλω σαν String τον αριθμό τηλεφώνου

"Phone number carrier: " + carrierName);//this.getCarrierName(phone.getPhoneNumber())); //καλώ τη μέθοδο geCcarrierName αυτή της κλάσης(με οτν τελεστή this) βάζοντας ως παράμετρο το getter getPhoneNumber() του αντικειμένου phone ώστε να στείλω σαν String τον αριθμό τηλεφώνου

PhoneOrderHandler.removePhone(phone);

PhoneOrderHandler.removeListener(this);

usePhone(phone);

}

}

**//Μέθοδος createPhoneSpec() της κλάσης PhoneShop**

//Όλα τα χαρακτηριστικά των τηλεφώνων πρέπει να παραχθούν με τυχαίο τρόπο

public Phone createPhoneSpec(){

//χρησιμοποίηση αντικειμένου random για την παραγωγή των τυχαίων χαρακτηριστικών

//οι τιμές θα κυμαίνονται μεταξύ της περιγραφής αντικειμένου όπως αναγράφεται στην κλάση Phone και Camera

//Το λογικό συμπέρασμα για την random είναι: το κατηγόρημα nextInt της random περιέχει το max μείον min όπου και δημιουργεί τη μέγιστη διαφορά

//στην οποία προστίθεται το min ώστε να δημιουργείται το ελάχιστο ζητούμενο και τέλος προστίθεται το +1 διότι η random μετράει από 0 εως το πλήθος -1

//παράδειγμα: min+(random.nextInt(max-min)+1)

int batterySize = 3000 + random.nextInt((5000-3000)+1);

//δημιουργία αντικειμένου screenSize. χρησιμοποίησα δύο μεταβλητές width και height όπου δημιουργήθηκαν με την random

Dimension screenSize = new Dimension();

int randomDimensionHeight = 500 + random.nextInt((3000-500)+1);

int randomDimensionWidth = 500 + random.nextInt((3000-500)+1);

screenSize.height = randomDimensionHeight;

screenSize.width = randomDimensionWidth;

//χρησιμοποιώ τη μέθοδο random ως όρισμα τη μέθοδο length του πίνακα και θα κάνω την έξοδο με array[randomManufacturers]

int randomManufacturers = random.nextInt(manufacturers.length);

//χρησιμοποιώ τη μέθοδο random ως όρισμα τη μέθοδο length του πίνακα και θα κάνω την έξοδο με array[randomOperatingSystem]

int randomOperatingSystem = random.nextInt(operatingSystems.length);

//Δημιουργία τυχαίου αριθού κινητού τηλεφώνου

String RandomMobileNumber = "+3069" + String.valueOf(random.nextInt(100000000));

//δημιουργία randoma storage

//String randomStorage = String.valueOf(2+random.nextInt((200-2)+1))+"GB";

int randomStorage = 2+random.nextInt((200-2)+1);

//δημιουργία random χαρακτηριστικά camera και δέσμευση χώρου και δημιουργία αντικειμένου camera

int randomCameraStats = 10 + random.nextInt((100-10)+1);

Camera camera = new Camera(randomCameraStats);

//Θα χρησιμοποιήσω πίνακα string έτσι ώστε να χρησιμοποιήσω την μέθοδο random και να παράγω δύο τυχαία τηλέφωνα featurephone ή smartphone

//δέσμευση στη μνήμη της κλάσης abstract και με μια if θα ελέγξω ποιο τηλέφωνο θα δημιουργηθεί

Phone phone = null;

String[] phoneRandomSelectionList = {"FeaturePhone","SmartPhone"};

int randomPhoneSelection = random.nextInt(phoneRandomSelectionList.length);

if(phoneRandomSelectionList[randomPhoneSelection].equals("FeaturePhone")){

//FeaturePhone featurePhone;

phone = new FeaturePhone(batterySize,screenSize,RandomMobileNumber,manufacturers[randomManufacturers],randomStorage);

phoneList.add(phone);

return phone;

}else {//if (phoneRandomSelectionList[randomPhoneSelection].equals("SmartPhone")){

phone = new SmartPhone(batterySize,screenSize,RandomMobileNumber,manufacturers[randomManufacturers],randomStorage,camera,operatingSystems[randomOperatingSystem]);

phoneList.add(phone);

return phone;

}

}//end of createPhoneSpec method

}//end of PhoneShop class

**//Μέθοδος usePhone της κλάσης Client**

/\*Η μέθοδος χρήσης του κινητού τηλεφώνου θα κάνει τα εξής:

Στην περίπτωση FeaturePhone θα καλεί τον αριθμό +30123456789

Στην περίπτωση SmartPhone Θα βγάζει μια φωτογραφία με φλας και ανάλυση 12 MP\*/

private void usePhone(Phone phone) {

//χρησιμοποίησα instanceof διότι θέλω να ελέξγω ότι το αντικείμενο να είναι στιγμιότυπο της κλάσης phone

if (phone instanceof FeaturePhone) {

phone.callNumber("+30123456789");

} else if (phone instanceof SmartPhone) {

//χρησιμοποιω downcasting ωστε να αποκτησω πρόσβαση στις μεθόδους της κλάσης SmartPhone https://www.baeldung.com/java-type-casting

//Casting από την υπερκλάση στην υποκλάση

((SmartPhone) phone).getCamera().setUseFlash(true);

((SmartPhone) phone).getCamera().setSelectedResolution(12);

((SmartPhone) phone).getCamera().takePicture();

}

}

**Ερώτημα Γ – Υλοποίηση προσομοίωσης με τα δεδομένα χρήστη**

Να συμπληρώσετε, όπου απαιτείται, τον κώδικα εντός της κλάσης Simulation με βάση τις απαιτήσεις/προδιαγραφές (απαίτηση/παραδοχή 1), τον κώδικα που έχετε λάβει, την εκτύπωση της **«Ενδεικτικής εκτέλεσης του προγράμματος»,** καθώς και τα σχόλια εντός των αρχείων κώδικα Java.

|  |
| --- |
| ***Μαθησιακά Αποτελέσματα***  Στην άσκηση 3.Γ. θα μάθετε:   * Να κάνετε προσομοίωση ζητώντας input από τον χρήστη * Να δημιουργείτε και να χειρίζεστε πίνακες ή/και λίστες αντικειμένων |

**Απάντηση**

*Εισάγετε τον κώδικα της κλάσης Simulation Java που φτιάξατε. Φροντίστε ο κώδικας να είναι μορφοποιημένος κατάλληλα και να είναι ευανάγνωστος. Θα πρέπει να* ***υπάρχει τεκμηρίωση με μορφή σχολίων****.*

*Εάν δεν έχετε δώσει απάντηση γράψτε με κεφαλαία γράμματα, ΔΕΝ ΑΠΑΝΤΗΘΗΚΕ.*

*Εάν εν γνώση σας δίνετε ελλιπή απάντηση γράψτε με κεφαλαία γράμματα, ΕΛΛΙΠΗΣ ΑΠΑΝΤΗΣΗ.*

***//Κλάση Simulation***

package eap.simulate;

import eap.observer.PhoneOrderHandler;

import eap.abstractfactory.FeaturePhone;

import eap.abstractfactory.Phone;

import eap.abstractfactory.SmartPhone;

import java.util.\*;

import java.util.List;

public class Simulation {

//Fixed array of Greek names

static final String[] names = {"George", "Maria", "Dimitris", "John", "Katerina",

"Panagiotis", "Konstantinos", "Christina", "Eleni", "Petros"};

// Στη μέθοδο main το πρόγραμμα αρχικά ζητά από τον χρήστη να δηλώσει τον αριθμό

// των κινητών τηλεφώνων που πρόκειται να δοθούν στη συνέχεια στη γραμμή παραγωγής,

// καθώς και τον αριθμό των πελατών που θα δημιουργηθούν

// και θα αναμείνουν για την ενδεχόμενη παραγωγή των τηλεφώνων που τους ενδιαφέρουν

public static void main(String[] args) {

//δημιουργία αντικειμένου random για την παραγωγή τυχαίων αριθμών της λίστας ονομάτων

Random random = new Random();

//Δημιουργία αντικειμένου scanner όπου καυ θα χρησιμοποιηθεί για είσομο του χρήστη και αντίστοιχων μεταβλητών όπου θα χρησιμοποιηθούν

//για την αποθήκευση της εισόδου του χρήστη

Scanner sc = new Scanner(System.in);

int userInputPhoneOrder=0, userInputClient=0;

//χρησιμοποίησα try catch ώστε να μπορέσω να δημιουργήσω μήνυμα προτροπής χρήσης μη αλφαριθμικών χαρακτήρων

try {

System.out.print("Please enter number of phones to be ordered: ");

userInputPhoneOrder = sc.nextInt();

//χρήση της do..while για αμυντικό προγραμματισμό της εισόδου του χρήστη με μηνύματα προτροπή λάθος χρήσης

do {

System.out.print("Please enter number of clients waiting for new phones (MAX:10): ");

userInputClient = sc.nextInt();

//χρήση επιλογής if..else of ώστε να μπορέσω να είμαι πιο συγκεκριμένος στο μήνυμα σφάλματος αμυνιτκού προγραμματισμού στον χρήστη

if (userInputClient < 0) {

System.out.println("\nYou 've entered wrong value you have enter: " + userInputClient + "clients which value is a negative number");

} else if (userInputClient > 10) {

System.out.println("\nYou have entered wrong value you have enter: " + userInputClient + "client which value is more than 10");

} else if (userInputClient == 0) {

System.out.println("\nYou 've entered wrong value you have enter: " + userInputClient + "clients. It needs at least 1 client");

}

} while (!(userInputClient >= 1 && userInputClient <= 10)); //θα βρίσκεται σε λούπα σε περίτωση ο χρήστης εισάγει κάτων του 1 και πάνω από 10 πελάτες

} catch (InputMismatchException e) {

System.out.println("Only arithmetic values are allowed. The program now...will exit");

System.exit(0); //χρήση System.exit(0) σε περίπτωση που ο χρήστης εισάγει κάτι διαφορετικό από αριθμητικούς χαρακτήρες να τερματίσει το πρόγραμμα

}

//έξοδος στην κονσόλα του προγράμματος, είναι ζητούμενο της άσκησης ανάλογα με την είσοδο του χρήστη.

System.out.println(userInputPhoneOrder+" phone orders have been placed by the PhoneShop!");

System.out.println(userInputClient+" clients are waiiting to buy a new phone!");

//ζητούμενο τον αριθμό clients που εισάγει ο χρήστης θα χρησιμοποιήσω την random ώστε να παράξω τυχαία τα ονόματα των πελατών.

//επίσης θα το προσθέσω μέσα σε μια for loop ώστε να δημιουργηθούν τόσα τηλέφωνα όσα δηλώσει ο χρήστης μέσω της μεταβλητής RandomUserCreation όπου ενημερώνεται μέσω του αντικειμένου Scanner

//final String[] PhoneListCustomerInterestedFor = {"FeaturePhone", "SmartPhone"};

//Δήλωση και αρχικοποίηση αντικεμένου PhoneShop

PhoneShop ps = new PhoneShop();

//Δήλωση και αρχικοποίηση αντικειμένου Phone

Phone phone = null; //αρχικοποίηση για χρήση του αντικειμένου έξω από την for..loop

//χρήση for..loop δημιουργία των τηλεφώνων ανάλογα με τη δήλωση της μεταβλητής που εισήγαγε ο χρήστης userInputPhoneOrder η οποία δημιουργήθηκε με το αντικείμενο scanner

for(int i =0;i<userInputPhoneOrder;i++){

phone = ps.createPhoneSpec();

PhoneOrderHandler.addPhone(phone);

}

//δημιουργία αντικειμένου Λίστας γενίκευσης Class για να αποθηκεύσω την επιλογή των τηλεφώνων που θέλουν οι clients μέσω του αντικειμένου random

ArrayList<Class> randomPhoneClientDesiresList = new ArrayList<Class>();

//Δήλωση και αρχικοποίηση μεταβλητής Class

//χρηση της random για την επιλογή τυχαίου τηλεφώνου FeaturePhone και SmartPhone στο οποίο αποθηκέυεται σε γενικευμένη μορφή .class

//στην οποία ο σκοπός δημιουργίας της είναι να σταλθεί στο αντικείμενο client που θα δημιουργηθεί στην συνέχεια

int randomPhoneClientDesires;

Class phoneClass = null; //αρχικοποίηση για χρήση του αντικειμένου έξω από την δομή επιλογής if..else

for(int i =0;i<=userInputClient;i++){

randomPhoneClientDesires = random.nextInt(2);

if(randomPhoneClientDesires == 0){

phoneClass = FeaturePhone.class;

randomPhoneClientDesiresList.add(phoneClass); //προσθήκη στην Λίστα randomPhoneClientDesiresList

}else if(randomPhoneClientDesires == 1){

phoneClass = SmartPhone.class;

randomPhoneClientDesiresList.add(phoneClass); //προσθήκη στην Λίστα randomPhoneClientDesiresList

}

}

//δημιουργία και αρχικοποίηση αντικειμένου client

Client client = null; //αρχικοποίηση για χρήση του αντικειμένου έξω από την for..loop

//με χρήση της for..loop θα δημιουργηθούν τόσοι πελάτες όσες δηλώσει ο χρήστης κατά την είσοδο του προγράμματος με χτήση της μεταβλητής userInputClient του αντικειμένου scanner

//χρησιμοποίησα το αντικείμενο random της κλάσης Random για την παραγωγή τυχαίων ονομάτων βάση του πίνακα String[] names.

for(int i =0; i<userInputClient;i++){

phoneClass = randomPhoneClientDesiresList.get(i); //Ανάγνωση από τη λίστα randomPhoneClientDesiresList με τη σειρά που έχει δημιουργηθεί για τον κάθε client και αποστολή τους ως όρισμα μέσω του ανρικειμένου client

int randomClientsCreationList = random.nextInt(names.length);

client = new Client(names[randomClientsCreationList], phoneClass); //αποστολή τω πληροφοριών στο αντικείμενο Client

PhoneOrderHandler.addListener(client); //αποστολή του αντικειμένου client στη μέθοδο addlistener της κλάσης PhoneOrderHandler

}

//εκκίνηση της κατασκευής των τηλεφώνων μέσω της μεθόδου buildPhones της κλάσης PhoneOrderHandler

PhoneOrderHandler.buildPhones();

//κάλεσμα μεθόδου report η οποία τυπώνει τους πελάτες οι οποίοι δεν έχουν λάβει τηλέφωνο.

//Η μέθοδος αυτή δημιουργήθηκε από εμένα.

reportGetSizeOfPhoneCreationListeners();

}

//μέθοδος report με το κάλεσμα της θα τυπώσει μέσω τις λίσταν πελατών της κλάσης PhoneCreationListener μιας και

//αυτή καλείται για να ενημερωθεί η λίστα πελατών phoneCreationListeners ώστε να υπολογίσω τους εναπομείναντες πελάτες οι οποίοι δεν έλαβαν τηλέφωνο

public static void reportGetSizeOfPhoneCreationListeners(){

PhoneOrderHandler phoneOrderHandler = new PhoneOrderHandler();

System.out.println("%%%%%----Report----%%%%%");

if(phoneOrderHandler.getSizeOfPhoneCreationListeners()==1){

System.out.println(phoneOrderHandler.getSizeOfPhoneCreationListeners()+" client did not get a phone..."); //πληθική συντακτική διόρθωση 1 client αντί 1 clients

}else if(phoneOrderHandler.getSizeOfPhoneCreationListeners()>1){

System.out.println(phoneOrderHandler.getSizeOfPhoneCreationListeners()+" clients did not get a phone...");

}else{

System.out.println("0 clients watting...all clients got their phones...");

System.out.println("%%%%%--------------%%%%%");

}

}

}

**Ερώτημα Δ – Χειρισμός αλφαριθμητικών και εκτύπωση σε αρχείο κειμένου**

Με βάση τις απαιτήσεις/προδιαγραφές, τον κώδικα που έχετε λάβει, καθώς και τα σχόλια εντός των αρχείων κώδικα Java, να υλοποιήστε τις παρακάτω μεθόδους:

1. Στην κλάση **Client** τις μεθόδους **“getCarrierName”** και **“checkNumberValidity”** (χειρισμός αλφαριθμητικών)
2. Στην κλάση **PhoneOrderHandler** τη μέθοδο “ **printCreatedPhoneSpecsToTextFile”** η οποία εκτυπώνει τα βασικά χαρακτηριστικά (μόνο από την κλάση Phone) όσων κινητών παράχθηκαν, στο αρχείο “output.txt”

|  |
| --- |
| ***Μαθησιακά Αποτελέσματα***  Στην άσκηση 3.Δ. θα μάθετε:   * Να γράφετε δεδομένα σε αρχεία, * Να χειρίζεστε αλφαριθμητικά |

**Απάντηση**

*Εισάγετε τον κώδικα των μεθόδων σε Java που φτιάξατε. Φροντίστε ο κώδικας να είναι μορφοποιημένος κατάλληλα και να είναι ευανάγνωστος. Στο φάκελο με το Project υπάρχουν τα αρχεία της βιβλιοθήκης GSON την οποία μπορείτε να χρησιμοποιήσετε προαιρετικά για να παράγεται το αποτέλεσμα. Εναλλακτικά μπορείτε να παράγεται το αποτέλεσμα με χρήση String τα οποία θα γράφετε στο αρχείο. Θα πρέπει να* ***υπάρχει τεκμηρίωση με μορφή σχολίων****.*

*Εάν δεν έχετε δώσει απάντηση γράψτε με κεφαλαία γράμματα, ΔΕΝ ΑΠΑΝΤΗΘΗΚΕ.*

*Εάν εν γνώση σας δίνετε ελλιπή απάντηση γράψτε με κεφαλαία γράμματα, ΕΛΛΙΠΗΣ ΑΠΑΝΤΗΣΗ.*

**//μέθοδος getCarrierName της κλάσης Client**

//Σύμφωνα με την περιγραφή πάνω από τον πίνακα carriers, η μέθοδος επιστρέφει είτε το όνομα του carrier, είτε Διαφημιστικά

public String getCarrierName(String phoneNumber) {

String tempPhoneNumber = phoneNumber.substring(3); //αφαίρεσα μέσω substring() τους τρεις πρώτους χαρακτήρες του διεθνή κωδικού σε temp μεταβλητή String

//Θα χρησιμοποιήσω την μέθοδο startsWith() της κλάσης String για να ελέγξω τις παραδοχές της άσκησης

if (tempPhoneNumber.startsWith("697") || tempPhoneNumber.startsWith("698") || tempPhoneNumber.startsWith("699")) {

return carriers[0];

} else if (tempPhoneNumber.startsWith("694") || tempPhoneNumber.startsWith("695") || tempPhoneNumber.startsWith("696")) {

return carriers[1];

} else if (tempPhoneNumber.startsWith("691") || tempPhoneNumber.startsWith("692") || tempPhoneNumber.startsWith("693")) {

return carriers[2];

} else{

return "Διαφημιστικά";

}

}

**//μέθοδος checkNumberValidity της κλάσης Client**

/\*Για να είναι έγκυρο ένα κινητό τηλέφωνο πρέπει να ισχύουν ταυτόχρονα τα εξής:

1. Να ξεκινάει από +3069 (είναι όλα από Ελλάδα και είναι όλα κινητά)

2. Στη συνέχεια να υπάρχουν ακριβώς 10 αριθμοί

3. Παραδοχή ότι δεν υπάρχουν κενά (white spaces μεταξύ των αριθμών)\*/

public boolean checkNumberValidity(String phoneNumber) {

//θα αφαιρέσω από τον αριθμό τηλεφώνου το πρώτο σύμβολο ώστε να μπορέσω να επεξεργαστώ μέσω του matcher τα αρχικά πρώτα ψηφία του αριθμού τηλεφώνου

Boolean internationalCodeNumberValidation = phoneNumber.startsWith("+30"); //Να ξεκινάει από +3069 (είναι όλα από Ελλάδα

//System.out.println(internationalCodeNumberValidation); //έλεγχος εξόδου το σβήνω μόλις τελειώσω

String tempPhoneNumber = phoneNumber.substring(3); //βγάζω από την αναζήτηση τον διεθνή κωδικό σε νέο String temp String

//System.out.println(tempPhoneNumber); //έλεγχος εξόδου το σβήνω μόλις τελειώσω

Boolean mobileNumberValidation = tempPhoneNumber.startsWith("69"); //Να ξεκινάει 69 όπου σημαίνει ότι όλα είναι κινητά

//System.out.println(mobileNumberValidation); //έλεγχος εξόδου το σβήνω μόλις τελειώσω

Boolean lengthNumberValidation = tempPhoneNumber.length() == 10; //έλεγχος να υπάρχουν ακριβώς 10 αριθμοί

//System.out.println(lengthNumberValidation); //έλεγχος εξόδου το σβήνω μόλις τελειώσω

//Εκκίνηση regex pattern

//Αρχικοποίηση αντικειμένου Pattern με χρήση της μεθόδου compile για να εισάγω την έκφραση regex

//Xρήση [0-9] για τον έλεγχο εύρους χαρακτήρων και χρηση [^\s] για τον έλεγχο της Παραδοχή της άσκησης ότι δεν υπάρχουν κενά μεταξύ αριθμών

//και τον τελεστή +. Επισυμάνω ότι για τον έλεγχο κενών χαρακτήρων μπορώ να χρησιμοποιήσω και το pattern [0-9] μιας και αν υπάρχει κενό δεν ανήκει στο range του pattern άρα θα βγάλει false

// \\+ για να εισάγο τον έλεγχο του χαρακτήρα + έπειτα χρησιμοποιώ range μέσα σε [] από το 0 έως το 9 και τέλος χρησιμοποιώ τνο τελεστή + για να κάνει έλεγχο

//το λιγότερο μια φορές

Pattern patternCheckDigits = Pattern.compile("\\+[0-9]+");

Matcher matcherCheckDigits = patternCheckDigits.matcher(phoneNumber);

//[^s]+$ μεταξύ αγκύλης γίνεται έλεγχος για whitespace (index 115) ο συμβολισμός + χρησιμοποιείται για την επανάληψη του pattern [^s] το λιγότερο μιαφορά και τέλος

//ο συμβολισμός $ γίνεται για να ελεγθεί μέχρι το τέλος της γραμμής

Pattern patternCheckWhiteSpace = Pattern.compile("[^s]+$");

Matcher matcherCheckWhiteSpace = patternCheckWhiteSpace.matcher(phoneNumber);

//Γίνεται έλεγχος για Whitespace ενα υπάρχει whiteSpace τότε τρέχω το replaceAll και βγάζω τα whitespace από τον αριθμό τηλεφώνου

if(matcherCheckWhiteSpace.matches()!=true){

tempPhoneNumber = tempPhoneNumber.replaceAll("\\s",""); //αν υπάρχουν whitespace τα αφαιρώ

}

if((internationalCodeNumberValidation == true) && (mobileNumberValidation == true) && (lengthNumberValidation == true) && (matcherCheckDigits.matches()==true) && (matcherCheckWhiteSpace.matches()==true) ){

return true;

}else{

return false;

}

}

}

**//** μέθοδος **printCreatedPhoneSpecsToTextFile τηε κλάσης PhoneOrderHandler**

//Η μέθοδος αυτή θα εκτυπώνει τα βασικά χαρακτηριστικά, μόνο από την κλάση Phone του κινητού τηλεφώνου

//όσων κινητών παράχθηκαν

private static void printCreatedPhoneSpecsToTextFile(List<Phone> phones){

//Αρχικοποιώ μετρητή counter τον οποίο θα τον χρησιμοποιήσω για μετρητή στην αποθήκευση του αρχείου

int counter = 1;

//χρησιμοποιώ try..catch με resourses για να κλείσει αυτόματα ο πόρος μετά το πέρας την εκτέλεσης του block

// και για να πιάσει τυχόν σφάλματα κατα τή δημιουργία ή/και εγγραφή του αρχείου.

//πηγή: https://docs.oracle.com/javase/tutorial/essential/exceptions/tryResourceClose.html

//δημιουργία και αρχικοποίηση αντικειμένου FileWriter ως όρισμα το όνομα της εξόδου του αρχείου

try(FileWriter fileWriter = new FileWriter("output.txt")){

//χρησιμοποιώ for..each loop για να κάνει itterate και εγγραφή όλων των στοιχείων από τη λίστα phones στο αρχείο output.txt

for(Phone output:phones){

fileWriter.write(counter+"."+"\n"+output.toString()+"\n"); //έξοδος της λίστας μέσω της μεθόδου write του αντικειμένου FileWriter

counter++; //μετρητής αύξησης κατά μία φορά μόλις ολοκληρωθεί το κάθε loop επιπλέον. το χρησιμοποιώ για την έξοδο αριθμοδότησης του αρχείου.

}

}catch(IOException e){

System.out.println(e);

}

}

**//Επιπλέον μέθοδος ως προς την υλοποίηση του προγράμματος την κλάσης PhoneOrderHandler:**

//Δημιουργία μεθόδου επιστροφής μέγεθος λίστας αναμένοντων πελατών όπου δεν έχουν πάρει τηλέφωνο

public int getSizeOfPhoneCreationListeners(){

return phoneCreationListeners.size();

}

**Υποδείξεις για τη συγγραφή της εργασίας**

1) Για την απάντηση της εργασίας θα πρέπει να χρησιμοποιηθεί το υπόδειγμα της εργασίας. Στο υπόδειγμα:

- Συμπληρώστε όλα τα στοιχεία με κίτρινο.

- Μην ξεχάσετε να δηλώσετε εάν η εργασία αποτελεί προϊόν αποκλειστικά δικής σας εργασίας.

- Ενσωματώστε τις απαντήσεις (διαγράμματα ή/και κώδικα) μετά το κάθε ερώτημα. Δεν θα πρέπει να κάνετε παραπομπές της μορφής «βλέπε αρχείο…».

- Αν δεν έχετε απαντήσει σε ένα ερώτημα γράψτε «**ΔΕΝ ΑΠΑΝΤΗΘΗΚΕ**».

- Αν απαντήσατε με ελλείψεις σε ένα ερώτημα γράψτε «**ΕΛΛΙΠΗΣ ΑΠΑΝΤΗΣΗ**».

2) Η συνεργασία στην ανάλυση της εργασίας επιτρέπεται, αλλά καλό είναι να αναφερθεί στον ειδικό χώρο στην πρώτη σελίδα της εργασίας. Η συνεργασία δεν πρέπει να οδηγεί σε από κοινού επίλυση και συγγραφή της εργασίας. Η υποβολή κοινών απαντήσεων από διαφορετικούς φοιτητές που συνεργάστηκαν δεν επιτρέπεται και θεωρείται ως **ΑΝΤΙΓΡΑΦΗ**. Οι απαντήσεις ελέγχονται, τόσο μεταξύ των φοιτητών του ιδίου τμήματος, όσο και μεταξύ φοιτητών διαφορετικών τμημάτων. Η αντιγραφή έχει ως αποτέλεσμα το **ΜΗΔΕΝΙΣΜΟ ΤΗΣ ΕΡΓΑΣΙΑΣ ΣΥΝΟΛΙΚΑ** και την παραπομπή των παραβατών στην Κοσμητεία της Σχολής Θετικών Επιστημών & Τεχνολογίας, σύμφωνα με τον εσωτερικό κανονισμό του ΕΑΠ.

3) Η εκπόνηση της εργασίας θα πρέπει να γίνει **αποκλειστικά και υποχρεωτικά** με το εργαλείο Visual Paradigm για τη UML και Netbeans για τη Java.

4) Η εργασία θα υποβληθεί στο σύστημα υποβολή εργασιών του study.eap.gr.

5) Ο φοιτητής θα πρέπει να στείλει την σε δύο αρχεία

* **Το 1ο αρχείο θα έχει όνομα PLH24\_2ERG\_EPITHETO\_ONOMA.doc** και είναι το παρόν αρχείο συμπληρωμένο με την απάντησή σας.
* **Το 2ο αρχείο είναι ένα συμπιεσμένο αρχείου zip ή rar** με όνομα PLH24\_2ERG\_EPITHETO\_ONOMA.<rar|zip>. Το συμπιεσμένο αρχείο θα πρέπει να αποτελείται από:

i. Ένα κατάλογο που θα περιέχει τα διαγράμματα που ζητά η κάθε άσκηση.

ii. Ο κατάλογος με τον κώδικα Java (άσκηση 3) θα πρέπει να περιλαμβάνει το project όπως αυτό δημιουργείται από το εργαλείο Netbeans και θα πρέπει να μπορεί να εκτελείται χωρίς αλλαγές από τον καθηγητή.

iii. Να γίνει χρήση λατινικών χαρακτήρων **ΑΠΟΚΛΕΙΣΤΙΚΑ** για την αποφυγή προβλημάτων με το moodle.

H εφαρμογή των παραπάνω κανόνων είναι **ΥΠΟΧΡΕΩΤΙΚΗ** και βαθμολογείται σύμφωνα με το αντίστοιχο κριτήριο αξιολόγησης. Η μη εφαρμογή **των παραπάνω κανόνων** μπορεί να οδηγήσει σε συνολική απόρριψη της εργασίας (π.χ. να δηλώσετε εάν η εργασία αποτελεί προϊόν αποκλειστικά δικής σας εργασίας, να μην αποστείλετε τον κώδικα σε netbeans, να μην υποβάλετε την εργασία ως doc αλλά ως pdf).

**ΚΑΛΗ ΕΠΙΤΥΧΙΑ!**