ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ
ΤΜΗΜΑ ΜΗΧ. Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ

ΜΥΥ301 ΑΝΑΠΤΥΞΗ ΛΟΓΙΣΜΙΚΟΥ **ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΗ ΑΣΚΗΣΗ**
ΟΚΤΩΒΡΙΟΣ 2023

Ημερομηνία Παράδοσης: 20-12-2023 **Π. Βασιλειάδης**

Η προγραμματιστική άσκηση για το μάθημα είναι **υποχρεωτική** και αφορά τη σχεδίαση, υλοποίηση και ρύθμιση ενός συστήματος λογισμικού. Η εργαστηριακή άσκηση προσφέρει **3 μονάδες** στον τελικό βαθμό του μαθήματος και εκπονείται σε ομάδες των **1 - 3 προσώπων**.

Φυσικά, πρέπει να πιάσετε τουλάχιστον τη βάση στην εργασία, όπως και στο διαγώνισμα. Σε περιπτώσεις εξαιρετικών εργασιών, η επίδοση επιβραβεύεται με bonus στον τελικό βαθμό.
Το σύστημα πρέπει να υλοποιηθεί σε όλα τα επί μέρους στάδια.

Η ανάλυση δεδομένων στο πλαίσιο της επιστήμης των δεδομένων (data science) απαιτεί απλές τεχνικές για να εξάγονται συμπεράσματα. Στο φετινό project θα εργαστούμε με δεδομένα φυσικών καταστροφών και θα προσπαθήσουμε να εξάγουμε συμπεράσματα ως προς τη συμπεριφορά του πραγματικού κόσμου, όπως αυτή εξάγεται δια των δεδομένων.

Θα δουλέψουμε με δεδομένα φυσικών καταστροφών από το IMF. Στο σχήμα φαίνεται μια απλοποιημένη μορφή των δεδομένων που χαρακτηρίζονται από (α) ένα αύξοντα αριθμό, (β) το κοινό όνομα μιας χώρας, (γ και δ) τους ISO2 (παλιός) && ISO3 (νέος) διεθνείς κωδικούς για τα ονόματα των χωρών, (ε) τον τύπο της φυσικής καταστροφής που καταγράφεται, και (στ) μια σειρά από μετρήσεις για τις χρονιές από το 1980 ως και σήμερα.

A	B	C	D	E	F	G	H	I	J	K	L	M
ObjectId	**Country**	**ISO2**	**ISO3**	**Indicator**	**1980**	**1981**	**1982**	**1983**	**1984**	**1985**	**1986**	**198**
334	Germany Dem Rep (for		DDR	TOTAL					2	1		
335	Germany Fed Rep (for		DFR	Flood				1				
336	Germany Fed Rep (for		DFR	Storm					2	1		
337	Germany Fed Rep (for		DFR	TOTAL				2	2	1		
338	Germany Fed Rep (for		DFR	Wildfire				1				
339	Ghana	GH	GHA	Drought	1							
340	Ghana	GH	GHA	Flood								
341	Ghana	GH	GHA	Storm								
342	Ghana	GH	GHA	TOTAL	1					1		
343	Ghana	GH	GHA	Wildfire						1		
344	Greece	GR	GRC	Drought								
345	Greece	GR	GRC	Extreme temperature						1		
346	Greece	GR	GRC	Flood								
347	Greece	GR	GRC	Storm								
348	Greece	GR	GRC	TOTAL				1	2	2		
349	Greece	GR	GRC	Wildfire				1	2	1		

Ουσιαστικά για κάθε συνδυασμό <χώρα, τύπος καταστροφής, χρονιά> μετράμε πόσες φορές συνέβη ένα καταστροφικό γεγονός.

Καλείσθε να κατασκευάσετε ένα σύστημα ανάλυσης των σχετικών δεδομένων φυσικών καταστροφών. Οι λειτουργίες που πρέπει να υποστηρίζονται είναι:

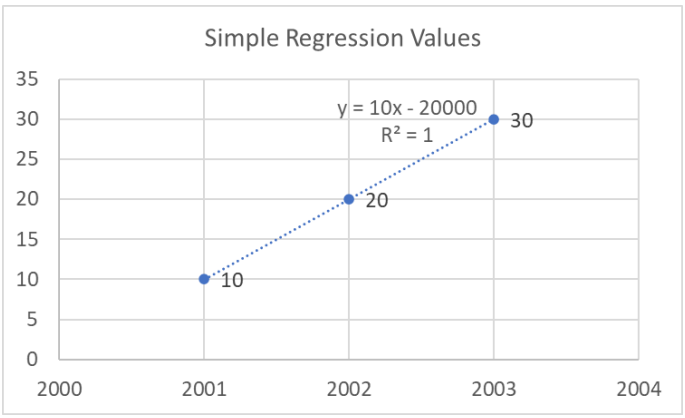
Φόρτωση από απλό κείμενο. Το σύστημα θα πρέπει να μπορεί να φορτώσει ένα αρχείο κειμένου με τις μετρήσεις στη δομή που προαναφέρεται. Το αρχείο που θα φορτωθεί θα είναι ένα απλό delimited αρχείο κειμένου (εδώ το δείχνουμε με formatted περιεχόμενο σαν xls για να φαίνεται λίγο πιο όμορφο). Μόλις φορτωθούν τα δεδομένα θα πρέπει κάθε γραμμή του αρχείου να αναπαρασταθεί από το κατάλληλο αντικείμενο (άρα πρέπει να ορίσετε τη σχετική κλάση), και όλα τα αντικείμενα αυτά να είναι μέρος της κατάλληλης συλλογής.

Ανάκτηση και παρουσίαση των στοιχείων για το συνδυασμό <χώρα, τύπος καταστροφής>. Ο αναλυτής πρέπει να μπορεί να προσδιορίσει το συνδυασμό χώρας και φυσικής καταστροφής, και το σύστημα να επιστρέψει στον αναλυτή τα σχετικά δεδομένα. Π.χ., αν ο αναλυτής ζητήσει τις πυρκαγιές για την Ελλάδα, το σύστημα θα πρέπει να του δώσει τα δεδομένα από την τελευταία γραμμή του παραπάνω σχήματος (349, Greece, GR, GRC, Wildfires, …).
Το αίτημα του αναλυτή, συνοδεύεται από ένα όνομα αιτήματος (π.χ., “Greece-Total”) για να κρατηθεί από το σύστημα στη συνέχεια.

Ανάκτηση και παρουσίαση των στοιχείων για το συνδυασμό <χώρα, τύπος καταστροφής, αρχικό έτος, τελικό έτος>. Ο αναλυτής θα πρέπει να μπορεί να προσδιορίσει το συνδυασμό χώρας και φυσικής καταστροφής, καθώς και το διάστημα [αρχικό έτος, τελικό έτος] και το σύστημα να ανακτήσει και να επιστρέψει στον αναλυτή τα σχετικά δεδομένα που αφορούν το συγκεκριμένο χρονικό εύρος. Π.χ., αν ο αναλυτής ζητήσει το σύνολο καταστροφών για την Ελλάδα στο χρονικό διάστημα [1980, 1983], το σύστημα θα πρέπει να του επιστρέψει, και επιδείξει, τα δεδομένα από την προτελευταία γραμμή του σχήματος για τα προαναφερθέντα έτη.
Το αίτημα του αναλυτή, συνοδεύεται από ένα όνομα αιτήματος (π.χ., "Greece-Wildfires-80_83") για να κρατηθεί από το σύστημα στη συνέχεια.

Υπολογισμός βασικών στατιστικών. Από τα αιτήματα που έχουν γίνει ήδη, ο αναλυτής δίνει το όνομα ενός από αυτά, και το σύστημα υπολογίζει βασικά περιγραφικά χαρακτηριστικά, όπως τη μέγιστη και ελάχιστη τιμή, των αριθμό των χρονιών που είχαν κάποιο συμβάν, τη μέση τιμή, την ενδιάμεση τιμή, και το συνολικό πλήθος συμβάντων (το άθροισμα των κελιών, δλδ.).

Υπολογισμός Regression. Η γραμμική παλινδρόμηση είναι η εξαγωγή μιας συνάρτησης της μορφής y = ax + b, που συσχετίζει δυο μεγέθη, τα x, y με μια γραμμική εξίσωση. Λέμε ότι το b είναι το intercept και το a είναι το slope. Επίσης, επειδή τα δεδομένα ποτέ δεν είναι ακριβώς πάνω στη γραμμή, υπάρχει και ένα λάθος που υπολογίζεται μαζί με την εξίσωση. Στο παράδειγμα του επόμενου σχήματος, στον οριζόντιο άξονα x είναι οι χρονιές και στον κάθετο άξονα y είναι τα συμβάντα που έλαβαν χώρα. Η γραμμή που φαίνεται να συνδέει τα σημεία στο σχήμα διέπεται από την εξίσωση y = 10x – 20000. Εδώ το -20000 είναι το intercept και το 10 είναι το slope. Εδώ, επειδή τα στοιχεία είναι φτιαγμένα επίτηδες έτσι, ο βαθμός επιτυχίας της φόρμουλας, που στο excel προκύπτει από τη μετρική coefficient of determination R^2 είναι 1.



Δε χρειάζεται να υλοποιήσετε εσείς κάποιο αλγόριθμο, σας δίνεται έτοιμη η βιβλιοθήκη apache commons math που υπολογίζει την παλινδρόμηση (και δίνει για μετρική λάθος μια άλλη μετρική, το slope error). Με βάση την κλίση της γραμμής, μπορούμε να κάνουμε και ένα χαρακτηρισμό για το πώς εξελίχθηκε η τάση των καταστροφών στη συγκεκριμένη χώρα για τη συγκεκριμένη περίοδο (ένα παράδειγμα είναι η διπλανή μέθοδος getLabel()).

```
public String getLabel() {
        if (Double.isNaN(slope))
                return "Tendency Undefined";
        else if(slope > 0.1)
                return "Increased Tendency";
        else if(slope < -0.1)
                return "Decreased Tendency";
        return "Tendency stable";
}
```

Αποθήκευση σε απλό κείμενο, html και markdown. Θέλουμε μια αναφορά για ένα αίτημα η οποία να μπορεί να αποθηκευθεί. Θα υποστηρίξουμε εναλλακτικούς τρόπους αποθήκευσης: (α) απλό tab-delimited κείμενο, (β) markdown και (γ) html format. Ο αναλυτής θα διαλέγει για ποιο από τα προηγούμενα αιτήματά τους θέλει αναφορά, και θα διαλέγει και τον τύπο της.
Για ένα text κείμενο, θέλουμε να καταγραφεί (α) το όνομα του αιτήματος, (β) το όνομα της χώρας και του του τύπου των καταστροφών, (γ) η λίστα των μετρήσεων, (δ) τα βασικά στατιστικά και (ε) το αποτέλεσμα του regression.
Για ένα markdown κείμενο, θέλουμε το αντίστοιχο, όπου επιπλέον: (α) με bold το όνομα του αιτήματος, (β) η χώρα και ο δείκτης σε italics και (β) τα στοιχεία των μετρήσεων σε πινακάκι. Δείτε για την πολύ απλή γλώσσα επισημείωσης markdown στο https://en.wikipedia.org/wiki/Markdown
Για ένα html κείμενο, αντίστοιχα με markdown. *Δείτε το παράδειγμα με την εξαγωγή φάσεων χρονοσειράς στο site του μαθήματος (παραδείγματα της ενότητας 6) για το πώς σχεδιάζεται και υλοποιείται μια τέτοια λύση.*

Έξοδος. Έξοδος από το πρόγραμμα. Ο χρήστης ερωτάται αν είναι σίγουρος ότι θέλει να φύγει από το πρόγραμμα και αν ναι, του κάνουμε τη χάρη.

Software Architecture & Specifications

Η αρχιτεκτονική του λογισμικού σε πακέτα, σας δίδεται εν μέρει προκαθορισμένα. Το υπό κατασκευή σύστημα οφείλει να είναι ένα **σύστημα 2 επιπέδων**, ενός **front-end client** κομματιού που είναι υπεύθυνο για την διάδραση με τον διαχειριστή ενός έργου και ενός **back-end server** κομματιού που είναι υπεύθυνο για την διεκπεραίωση των use cases που προκύπτουν από την αρχική περιγραφή της λειτουργικότητας του συστήματος.

Α. Στην back-end πλευρά του server οφείλουν να υπάρχουν διάφορα packages (πακέτα), έκαστο με τη δική του λειτουργικότητα. Στην υλοποίηση που έκανα εγώ για να ελέγξω την εκφώνηση, τα πακέτα που έβαλα στο back-end είναι:

- Ένα πακέτο για την φιλοξενία του κεντρικού controller (που υποστηρίζει τη διεκπεραίωση κάθε use case μέσω της σχετικής μεθόδου).
- Ένα πακέτο για την υπηρεσία ανάκτησης από αποθηκευμένο αρχείο.
- Ένα πακέτο για την υπηρεσία αναλυτικής επεξεργασίας των δεδομένων.
- Ένα πακέτο για την υπηρεσία παραγωγής αναφορών.
- Ένα πακέτο για την φιλοξενία των domain classes.
- Ένα πακέτο για την φιλοξενία των interfaces δια των οποίων επικοινωνεί το front-end με το back-end

Β. Στη front-end πλευρά, σας δίνεται ένα πακέτο για να στεγάσει τη αλληλεπίδραση του συστήματος με τον project manager. Εκεί υπάρχει αφενός ένα απλό console-based παράδειγμα, ή ένα Graphical User Interface, καθώς και οι βοηθητικές κλάσεις. Αν υλοποιηθεί σωστά το back-end, το front-end παίζει μια χαρά.

Περιορισμοί:
Είναι υποχρεωτικό και απαράβατο, ΝΑ ΜΗΝ ΑΛΛΑΞΕΤΕ τα interfaces που σας δίνονται ως μέρος της εκφώνησης, αλλά να τα υλοποιήσετε επακριβώς. Το ποιες κλάσεις θα εντάξετε μέσα στο κάθε πακέτο είναι δικό σας θέμα και αντικείμενο της σχεδίασης, υλοποίησης και ελέγχου που θα κάνετε, καθώς και της αξιολόγησής τους. Έχετε δικαίωμα να υλοποιήσετε τις κλάσεις που λείπουν στο εσωτερικό των πακέτων με όποιο τρόπο θέλετε εσείς.
Είναι υποχρεωτικό και απαράβατο, ΝΑ ΚΑΤΑΣΚΕΥΑΣΤΟΥΝ UNIT TEST (Happy Day, Rainy Day) για όλα τα use cases που θα εντοπίσετε. Προφανώς tests πρέπει να υπάρχουν και για τις επί μέρους domain/bridge κλάσεις αλλά το βασικό είναι να ελεγχθούν τα use cases!
Χρησιμοποιήστε Junit 4 και όχι 5 (για να έχουμε ένα κοινό setup για να ελέγξουμε τα projects).

Good Practice:
Εσωτερικά στο σύστημα, για να επικοινωνήσει ο κεντρικός controller που θα βρίσκεται στο πακέτο engine με τα υποσυστήματα που θα υλοποιούν τα διάφορα services, είναι χρήσιμο τα πακέτα των υποσυστημάτων να «εξάγουν» προς το υπόλοιπο σύστημα ένα interface και ένα factory (θα πούμε τι είναι αυτά στο μάθημα). Αυτό βοηθά στην απομόνωση του εσωτερικού των πακέτων από τα υπόλοιπα πακέτα και βελτιώνει τη συντηρησιμότητα του κώδικα.
Κάποιες domain classes αναγκαστικά θα υλοποιήσουν τα interfaces του πακέτου dom2app.

Η κεντρική ιδέα. Σας δίνεται ένα πακέτο app (με υποπακέτα) με τους εξής 2 τρόπους λειτουργίας: ένα naïve client που πρακτικά λειτουργεί εν είδη τεστ, και (β) ένα graphical user interface πλήρως υλοποιημένο (ναι, για όσους θέλουν να το ψάξουν περαιτέρω). Τα κομμάτια αυτά όλα επικοινωνούν με το back-end μέσω μιας γέφυρας από 3 μέρη: (α) τον *AppController*, που κάνει τη δουλειά να υποδέχεται τα αιτήματα των χρηστών από τις προαναφερθείσες boundary classes (το GUI και το naïve client) τις οποίες να ζητά να διεκπεραιωθούν, (β) από το back-end που εκπροσωπείται από το *IMainController* το οποίο είναι μια βιτρίνα που κρύβει όλο το back-end, (γ) και του οποίου οι μέθοδοι επιστρέφουν αντικείμενα από κλάσεις που περιέχονται στο πακέτο *dom2app*, τις οποίες ξέρουν και το front- και το back-end.
Οτιδήποτε βρίσκεται μετά το back-end, ξεκινώντας με την υλοποίηση του *IMainController* και το σχετικό factory (αυτό που σας εδόθη είναι το μόνο σημείο στον κώδικα που θέλει διόρθωση) είναι προς σχεδίαση, υλοποίηση και έλεγχο από εσάς.

Υλικό και οδηγίες

Υλικό. Όλο το υποστηρικτικό υλικό για το project θα βρίσκεται στο URL
https://www.cs.uoi.gr/~pvassil/courses/sw_dev/exercises/supportingMaterial/2023-2024/
όπου θα βρείτε ένα αρχείο .zip με το σκελετό ενός Eclipse Java project με
(α) τη δομή του src, σημαντικό κομμάτι του κώδικα στο front-end, και τα σχετικά interfaces/classes of the back-end/front-to-back-bridges,
(β) το βασικό αρχείο με δεδομένα καθώς και πιο μικρά αρχεία για να κάνετε ελέγχους και για να τρέξετε το σύστημα που θα φτιάξετε (src/test/resources/input), καθώς και το πώς αναμένεται να είναι το περιεχόμενο των reports (src/test/resources/output),
(γ) ένας φάκελος libs με τα σχετικά jar (α) για τη ζωγραφική στο front-end και (β) για να αξιοποιήσετε για το analysis (δείτε στο Apache commons για descriptive stats && simple regression https://commons.apache.org/proper/commons-math/userguide/stat.html).

Υπόδειγμα αναφοράς: για τα επιμέρους στάδια, μπορείτε να συμπληρώνετε / αναθεωρείτε σταδιακά την αναφορά σας. Για διευκόλυνσή σας, υπάρχει ένα υπόδειγμα στο http://www.cs.uoi.gr/~pvassil/courses/sw_dev/exercises/TemplateFinalReport.zip .
ΥΠΟΧΡΕΩΤΙΚΑ ΠΡΕΠΕΙ ΝΑ ΑΚΟΛΟΥΘΗΣΕΤΕ ΤΟ ΥΠΟΔΕΙΓΜΑ ΠΟΥ ΣΑΣ ΔΙΝΕΤΑΙ!

Important ToDo. Επιπλέον όλων, εσείς πρέπει:
- Να μετονομάσετε το δοθέν Eclipse project ώστε στο όνομα που θα έχει, τα "AM1", "AM2", "AM3" να αντικατασταθούν από τους συγκεκριμένους Αρ. Μητρώου των μελών της ομάδας, με αύξουσα σειρά, (ώστε αφού τα κάνετε turnin να ξέρουμε ποιανού είναι κάθε project). Π.χ., αν στην ομάδα μετέχουν οι φοιτητές με ΑΜ 345, 344, 567, το project υποχρεωτικά πρέπει να ονομάζεται ως: **`344_345_567_NaturalDisasters`** (Στο Eclipse, σχεδόν τα πάντα μετονομάζονται με δεξί κλικ -> Refactor -> Rename). Στο αρχειάκι .project που συνοδεύει το Eclipse project θα πρέπει να μετονομάσετε το project επίσης.
- Να προσθέσετε ένα αρχείο κειμένου Readme.txt που θα λέει (α) ονόματα και ΑΜ, (β) αν τυχόν χρειάζεται, τι απαιτείται για να τρέξει το πρόγραμμά σας χωρίς προβλήματα στον έλεγχο από τους βοηθούς (π.χ., κάποιες ειδικές βιβλιοθήκες που τυχόν χρησιμοποιήσατε)

Στη διάρκεια του εξαμήνου:
- Σίγουρα θα δοθούν περαιτέρω εξηγήσεις στην εκφώνηση και ενδεχομένως να αλλάξει/εμπλουτισθεί κάποιο μέρος της εκφώνησης! (άρα το δημοσιευθέν αρχείο της εκφώνησης μπορεί να αλλάζει).
- **Αργότερα** στο εξάμηνο, **θα σας ζητηθεί να εγγράψετε την ομάδα σας** σε μία φόρμα εγγραφής. Ομάδες που δεν εγγραφούν κινδυνεύουν να μην βαθμολογηθούν. Θα δοθούν οδηγίες αφού έχουμε μια πιο καθαρή εικόνα του τι γίνεται με το εξάμηνο.

Παραδοτέα

Θα κάνετε turnin 2 αρχεία:
- a single, all-encompassing zip file with the code for all classes (λόγω μεγέθους βγάλτε από το zip σας το φάκελο libs, για να μπορέσει να περάσει τα όρια μεγέθους του turnin).
- the report (pdf) with all the design and the documentation of the project.

Χρονοδιάγραμμα

Η προθεσμία είναι στις 20/12/2023. Η ΠΡΟΘΕΣΜΙΑ ΤΗΣ ΠΑΡΑΔΟΣΗΣ ΕΙΝΑΙ ΑΝΕΛΑΣΤΙΚΗ!
Δεν θα ξεπεράσουμε το όριο των Χριστουγέννων. Η πράξη έχει αποδείξει ότι στις γιορτές οι ομάδες αποσυντονίζονται σε πολύ μεγάλο βαθμό. Θα πιεστείτε περισσότερο πριν τις γιορτές, αλλά θα φύγετε για τις γιορτές χωρίς το φορτίο του project.

ΚΑΛΗ ΕΠΙΤΥΧΙΑ!