ΓΙΑ ΤΟΥΣ ΠΑΛΙΟΥΣ: έχω ξαναδώσει στο παρελθόν project με Gantt diagrams, που ήταν πολύ διαφορετικό. Μην επαναπαυθείτε: **η φετινή εκφώνηση είναι πολύ διαφορετική**!

ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ
ΤΜΗΜΑ ΜΗΧ. Η/Υ & ΠΛΗΡΟΦΟΡΙΚΗΣ

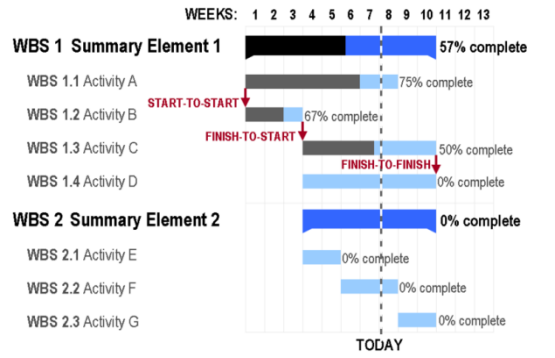
ΜΥΥ301 / ΠΛΥ 308 ΑΝΑΠΤΥΞΗ ΛΟΓΙΣΜΙΚΟΥ **ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΗ ΑΣΚΗΣΗ**
ΟΚΤΩΒΡΙΟΣ 2024

Ημερομηνία Παράδοσης: 22-12-2024, 23.59 **Π. Βασιλειάδης**

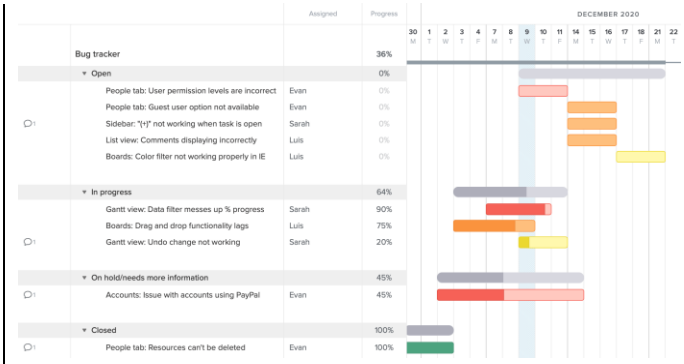
Η προγραμματιστική άσκηση για το μάθημα είναι **υποχρεωτική** και αφορά τη σχεδίαση, υλοποίηση και ρύθμιση ενός συστήματος λογισμικού. Η εργαστηριακή άσκηση προσφέρει **3 μονάδες** στον τελικό βαθμό του μαθήματος και εκπονείται σε ομάδες των **1 - 3 προσώπων**.

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.

Ένα *έργο* (project) είναι μια οργανωμένη προσπάθεια που έχει σχεδιασθεί κατάλληλα, με σκοπό την υλοποίηση ενός καλά ορισμένου στόχου, που μπορεί να είναι ένα νέο προϊόν, υπηρεσία, ή αποτέλεσμα. Λόγω πολυπλοκότητας, βασική αρχή της διαχείρισης έργων είναι η κατάτμησή τους σε επί μέρους εργασίες και ο χρονοπρογραμματισμός τους. Ένα βασικό εργαλείο διαχείρισης έργων είναι τα διαγράμματα Gantt. Δείτε δύο παραδείγματα διαγραμμάτων Gantt.



https://en.wikipedia.org/wiki/Gantt_chart



https://www.teamgantt.com/blog/gantt-chart-example

Η βασική ιδέα σε ένα διάγραμμα Gantt είναι ότι χρησιμοποιούμε *εργασίες* (tasks) ως το βασικό δομικό υλικό ενός έργου. Μία εργασία είναι μια συμπαγής δράση που έχει ένα συγκεκριμένο στόχο στο πλαίσιο του έργου: είναι ένα «βήμα» προς τον τελικό στόχο. Οι εργασίες έχουν ένα ακέραιο αριθμό που τις χαρακτηρίζει και ένα κείμενο που περιγράφει την ουσία τους. Επίσης έχουν ημερομηνία έναρξης, ημερομηνία λήξης (και κατά συνέπεια μια διάρκεια), κόστος χρηματικό και ένα απαιτούμενο φόρτο εργασίας.

Οι εργασίες είναι είτε απλές είτε σύνθετες. Οι *απλές* εργασίες έχουν μια ημερομηνία έναρξης, μια ημερομηνία λήξης, ένα κόστος και ένα φόρτο εργασίας. Οι *σύνθετες*, ή *κορυφαίου επιπέδου* (top level) εργασίες (α) αποτελούνται από άλλες εργασίες, τις οποίες λέμε *υποεργασίες*, και (β) έχουν τα ίδια στοιχεία με τις απλές, αλλά αυτά υπολογίζονται ως εξής: (α) η μικρότερη από τις ημερομηνίες έναρξης των υποεργασιών είναι η έναρξη της σύνθετης, (β) η μεγαλύτερη από τις ημερομηνίες λήξης των υποεργασιών είναι η λήξη της σύνθετης, (γ) το κόστος της σύνθετης εργασίας είναι το άθροισμα των επί μέρους κοστών των υποεργασιών της και (δ) αντιστοίχως για το φόρτο εργασίας.
Κάνουμε την *απλοποιητική υπόθεση εργασίας ότι οι σύνθετες εργασίες έχουν μόνο απλές υποεργασίες (δλδ., δε θα τύχει μια σύνθετη εργασία να έχει στις υποεργασίες της μια άλλη σύνθετη).*

Καλείσθε να κατασκευάσετε ένα σύστημα διαχείρισης διαγραμμάτων Gantt και παραγωγής, ως output, αρχείων Excel. Οι λειτουργίες που πρέπει να υποστηρίζονται είναι:

ΦΟΡΤΩΣΗ
Φόρτωση από αρχείο εισόδου. Το σύστημα θα πρέπει να μπορεί να φορτώσει μια αποθηκευμένη αναπαράσταση ενός Gantt diagram από ένα αρχείο εισόδου. Το αρχείο που θα

φορτωθεί θα μπορεί να ανήκει σε μια από τις επόμενες κατηγορίες: (α) ένα απλό delimited αρχείο κειμένου, (β) ένα αρχείο Excel τύπου xls, και (γ) ένα αρχείο Excel τύπου xlsx (θέλουμε να μπορείτε να υποστηρίξετε ΚΑΙ ΤΙΣ ΤΡΕΙΣ κατηγορίες). Εδώ δείχνουμε ένα παράδειγμα με formatted περιεχόμενο για να εξηγήσουμε τι αναμένεται να βρούμε σε ένα αρχείο εισόδου.

TaskId	TaskText	MamaId	Start	End	Cost	Effort
100	***Prepare a shopping list***	***0***				
101	Visit all the closets to see what's missing	100	1	5	50	5
102	Write down what you need to buy	100	1	6	10	6
200	***Proceed to the super market***	***0***				
201	Drive the car	200	7	9	10	5
300	***Buy and pay***	***0***				
307	Put stuff in the market basket	300	9	15	40	12
302	Pay at the cashier and leave	300	15	28	442	1

Κάθε εργασία είναι μία γραμμή του αρχείου. Οι στήλες χωρίζονται με ένα διαχωριστικό string, π.χ., ένα tab. Κάθε εργασία πρέπει να έχει ένα μοναδικό TaskId το οποίο δεν μπορεί να το έχει άλλη. Αν μια εργασία είναι top-level, το πεδίο mamaId είναι 0 (ναι, απαγορεύεται να υπάρχει εργασία με taskId == 0, το 0 στο mamaId είναι μια σύμβαση για να μας πει ότι η εργασία είναι κορυφαίου επιπέδου). Αν μια εργασία είναι σύνθετη, ΔΕΝ περιλαμβάνει στοιχεία για ημερομηνίες, κόστος φόρτο (θα υπολογισθούν από τα συστατικά της). Στο παραπάνω παράδειγμα:

- Η αρχική μπλε header γραμμή είναι απλά για να δείξει εδώ τι πεδία αναμένεται να έχουν οι εργασίες – θα λείπει από το αρχείο σας.
- Οι τοπ εργασίες είναι με bold, italics (100, 200, 300)

Είναι σημαντικό ότι το αρχείο μπορεί να περιέχει τις εργασίες με οποιαδήποτε σειρά. Εσείς αφού τις φορτώσετε, θα πρέπει να έχετε κρατήσει κάπου την λίστα όλων των εργασιών ταξινομημένη ως εξής:

- Όλες οι top-level εργασίες να είναι ταξινομημένες ως προς το start χρονικό σημείο τους, και αν υπάρχουν ισοπαλίες, να προηγείται αυτή με το μικρότερο taskId.
- Ανάμεσα στη top-level εργασία $task_i$ και την επόμενη top-level εργασία $task_{i+1}$, μπαίνουν οι υποεργασίες της $task_i$ (αν υπάρχουν), που ταξινομούνται ομοίως: αρχικά με βάση το start και αν υπάρχει ισοπαλία με βάση το taskId.

Προφανώς, άμα η συνολική λίστα είναι ταξινομημένη όπως προαναφέρθηκε, αν ζητηθούν μόνο οι top-level εργασίες, είναι κι αυτές ταξινομημένες.

Στο παρακάτω παράδειγμα το αρχείο **δεν** είναι ταξινομημένο σωστά: η 104 θα πρέπει να πάει μετά την 105, λόγω του ότι ξεκινά πιο μετά! (ασχέτως που έχει πιο μικρό TaskId δλδ).

TaskId	TaskText	MamaId	Start	End	Cost	Effort
100	***Prepare Fry***	***0***				
101	Turn on burner (low)	100	1	1	10.0	15.0
102	Break eggs and pour into fry	100	2	4	10.0	14.0
103	Steer mixture to avoid sticking	100	5	10	10.0	14.0
104	Throw yellow cheese into fry	100	**6**	12	10.0	16.0
105	Salt, pepper	100	**5**	5	10.0	17.0
106	Turn burner off	100	12	12	10.0	15.0
200	***Prepare the bread***	***0***				
201	Heat bread in toaster	200	10	12	10.0	28.0
202	Little bit of salt, galric spice to bread	200	12	12	10.0	28.0
300	***Serve eggs***	***0***				
301	Put bread in plate	300	13	13	10.0	50.0
302	Put eggs on bread	300	14	14	10.0	50.0
303	Wash fry	300	15	20	10.0	49.0

ΦΙΛΤΡΑ ΑΝΑΚΤΗΣΗΣ (ΥΠΟ)ΔΙΑΓΡΑΜΜΑΤΩΝ GANTT

Ανάκτηση πλήρως όλων των εργασιών. Για κάθε εργασία επιστρέφονται όλα τα στοιχεία τους: id, περιγραφή, αρχή, τέλος, κόστος, φόρτος. Προφανώς πρέπει, με ενιαίο τρόπο να μπορούν να ανακτηθούν όλα τα στοιχεία, είτε δόθηκαν, είτε προκύπτουν από υπολογισμό (π.χ., το κόστος μιας

σύνθετης εργασίας). Το αποτέλεσμα είναι ταξινομημένο (αν η ταξινόμηση γίνει κατά τη φόρτωση, είναι έτοιμη).

Ανάκτηση μόνο των εργασιών κορυφαίου επιπέδου. Από όλες τις εργασίες, επιστρέφουμε μόνο οι εργασίες κορυφαίου επιπέδου. Αντίστοιχο με το προηγούμενο, πάλι ταξινομημένο.

Ανάκτηση μόνο των εργασιών εντός ενός εύρους. Ανακτώνται ταξινομημένες, μόνο οι εργασίες που το TaskId τους είναι μεγαλύτερο ίσο από κάποιο κάτω όριο και μικρότερο ή ίσο από κάποιο άνω όριο, ασχέτως από το αν είναι κορυφαίου ή μη επιπέδου. Πάλι ταξινομημένες.

ΠΑΡΑΓΩΓΗ OUTPUT

Προετοιμασία αρχείου εξόδου. Ο διαχειριστής έργων προσδιορίζει το αρχείο στο οποίο θα καταγράφονται αποτελέσματα από τα παραπάνω φίλτρα. Ο τύπος του αρχείου μπορεί να είναι xls ή xlsx και πρέπει να προσδιοριστεί και το μονοπάτι στο οποίο (θα) βρίσκεται το αρχείο. Το αρχείο μπορεί να υπάρχει ήδη (οπότε θα του προστεθούν νέα φύλλα, ή θα «πανωγραφούν» υπάρχοντα) ή να είναι νέο, οπότε θα προσθέσουμε νέα φύλλα εργασίας (Sheet).

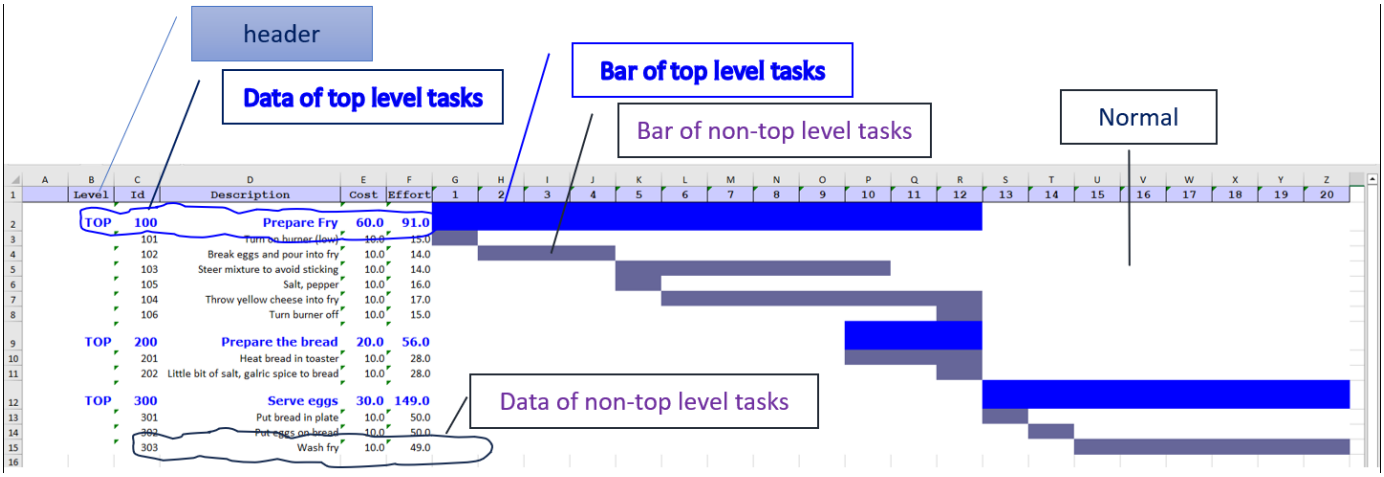
Κατασκευή ενδιάμεσης αναπαράστασης και παραγωγής του σχετικού φύλλου. Η λειτουργία αυτή γίνεται για να σας δείξει την αξία των ενδιάμεσων αναπαραστάσεων και για τον έλεγχο, και για την κατασκευή των τελικών αναπαραστάσεων για το τελικό output.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1		Level	Id	Description	Cost	Effort	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
2		TOP	100	Prepare Fry	60.0	91.0	x	x	x	x	x	x	x	x	x	x	x	x								
3			101	Turn on burner (low)	10.0	15.0	x																				
4			102	Break eggs and pour in	10.0	14.0		x	x	x																	
5			103	Steer mixture to avoid	10.0	14.0					x	x	x	x	x	x											
6			105	Salt, pepper	10.0	16.0					x																
7			104	Throw yellow cheese in	10.0	17.0						x	x	x	x	x	x	x									
8			106	Turn burner off	10.0	15.0												x									
9		TOP	200	Prepare the bread	20.0	56.0										x	x	x									
10			201	Heat bread in toaster	10.0	28.0										x	x	x									
11			202	Little bit of salt, galric s	10.0	28.0												x									
12		TOP	300	Serve eggs	30.0	149.0													x	x	x	x	x	x	x	x
13			301	Put bread in plate	10.0	50.0													x							
14			302	Put eggs on bread	10.0	50.0														x						
15			303	Wash fry	10.0	49.0															x	x	x	x	x	x

Η ενδιάμεση αναπαράσταση, αναπαριστά κάθε task σαν μια λίστα/πίνακα από συμβολοσειρές. Από αυτές τις συμβολοσειρές ενός task: η πρώτη είναι κενή, η δεύτερη λέει TOP για τις εργασίες κορυφαίου επιπέδου και τίποτα για τις άλλες, οι επόμενες δύο έχουν το κόστος και την προσπάθεια του task, και μετά, για κάθε χρονική στιγμή μεταξύ start και end, όπου το task είναι ενεργό, η σχετική συμβολοσειρά περιέχει ένα 'x'.
Όταν εξάγουμε την ενδιάμεση αναπαράσταση σε ένα φύλλο εργασίας, πρέπει να καταγράψουμε κάθε task σαν μία γραμμή του φύλλου εργασίας, αξιοποιώντας τη μετατροπή σε συμβολοσειρές.
Επίσης, στην αρχή (γραμμή 1 του φύλλου εργασίας) υπάρχει και το σχετικό header row, που εξηγεί τι είναι κάθε στήλη. Δε χρειάζεται κανένα formatting.

Κατασκευή νέου Formatting Style. Η μορφοποίηση του φύλλου εργασίας μπορεί να γίνει με πολλά διαφορετικά στυλ. Αυτό αφορά και τις συμβολοσειρές που χρησιμοποιούνται για τα διάφορα κομμάτια του φύλλου εργασίας, και τις μπάρες, και τα άδεια κελιά. Για τις συμβολοσειρές, οι οποίες αφορούν τα γράμματα στα κομμάτια κειμένου ενός φύλλου εργασίας, θέλουμε προσδιορισμό για γραμματοσειρά, μέγεθος, αν θα είναι bold, italic, strikeout, στοίχιση (αριστερά, δεξιά, κέντρο, ζυγισμένη) και χρώμα. Για τις μπάρες που θα χρησιμοποιηθούν για να δείξουν τη διάρκεια κάθε task θέλουμε χρώμα, αν το γέμισμα θα είναι ανύπαρκτο, συμπαγές, ή διάστικτο και αν το κείμενο περιελίσσεται μέσα σε ένα κελί ή όχι (text wrapping). Κάθε στυλ έχει και το μοναδικό όνομά του για να ξεχωρίζουν μεταξύ τους.
Έτσι, μπορεί, για παράδειγμα, να έχουμε μεγαλύτερα (1) μπλε bold γράμματα για τις κορυφαίες εργασίες, και (2) μικρότερα για τις μη κορυφαίες, καθώς και (3) χαρούμενες μπλε μπάρες για τις κορυφαίες εργασίες, και (4) πιο άτονες, γκρι για τις μη κορυφαίες (το χρώμα της μπάρας προκύπτει από το σχετικό γέμισμα), (5) μωβ γέμισμα για την γραμμή τίτλων, και (6) άσπρο γέμισμα για όποιο άλλο κελί.
Θα χρειαστείτε (α) το σύστημά σας, να φτιάχνει ήδη από μόνο του μια συλλογή από στυλ (StyleGallery) που να παρέχει τουλάχιστον τα εξής στυλ: "Normal", "DefaultHeaderStyle", "TopTask_bar_style", "TopTask_data_style", "NonTopTask_bar_style", "NonTopTask_data_style" (τα ονόματα εξηγούν και πού χρησιμοποιείται το κάθε στυλ by default), και, (β) να μπορείτε να προσθέσετε και νέα στυλ μέσω της παρούσας λειτουργικότητας.

Κατασκευή νέου Sheet. Για να φτιάξουμε ένα νέο φύλλο εργασίας, πρέπει να προσδιορίσουμε (α) το όνομά του, (β) το ποιες υποεργασίες θα αναπαραστήσει ως διάγραμμα Gantt (οι οποίες μπορεί να προκύψουν από την κλήση ενός από τα παραπάνω 3 φίλτρα) και (γ) προδιαγραφή των ειδών «στυλ μορφοποίησης» για τα κελιά του διαγράμματος Gantt, και συγκεκριμένα: headerStyleName για την γραμμή τίτλων, topBarStyleName, για την μπάρα για μια εργασία κορυφαίου επιπέδου, topDataStyleName για τα αντίστοιχα δεδομένα, nonTopBarStyleName και nonTopDataStyleName για μη-κορυφαίες εργασίες αντίστοιχα, και τέλος normalStyleName για ένα γενικό πρότυπο διαμόρφωσης κελιών.



Παρατηρήστε την σχετική εικόνα. Έχουμε πάρει το παράδειγμα του έργου τηγανίσματος αυγών και έχουμε επιλέξει όλες τις εργασίες του έργου. Κατ' αρχήν υπάρχει μια μωβ header γραμμή που λέει (α) για τα μεν δεδομένα, τι είναι η κάθε στήλη, και (β) για τη διάρκεια του έργου, από μία στήλη για κάθε χρονική μονάδα του έργου, ξεκινώντας από 1. Για κάθε task υπάρχει μία γραμμή στο φύλλο εργασίας, όπου στην αρχή αναγράφονται τα στοιχεία και στη συνέχεια χρωματίζεται κατάλληλα ένα κελί που περιέχει τον χαρακτήρα ' ', ανάλογα με το στυλ διαμόρφωσης του κελιού. Παρατηρήστε πώς οι εργασίες κορυφαίου επιπέδου διαμορφώνονται διαφορετικά από τις εργασίες μη κορυφαίου επιπέδου.
Hint: Είναι πολύ βολικό να αξιοποιήσετε τη σχετική ενδιάμεση αναπαράσταση για να δημιουργήσετε το επιθυμητό αποτέλεσμα.

Χρήσιμες βιβλιοθήκες και εργαλεία

Το Apache POI (https://poi.apache.org/), ένα έργο του Apache Software Foundation, παρέχει βιβλιοθήκες Java για ανάγνωση και εγγραφή αρχείων σε μορφές του Microsoft Office, όπως Word, PowerPoint και Excel. Εμάς μας ενδιαφέρουν οι βιβλιοθήκες για το Excel (https://poi.apache.org/components/spreadsheet/index.html) με την XSSF για αρχεία xlsx και την HSSF για αρχεία xls.

Μια αρχική εισαγωγή θα βρείτε στο
https://poi.apache.org/components/spreadsheet/quick-guide.html

Δείτε (οπωσδήποτε) παραδείγματα στο
https://poi.apache.org/components/spreadsheet/examples.html
και ιδίως το business plan (!!) και το calendar

Επίσης μπορείτε να δείτε και τα
https://mkyong.com/java/apache-poi-reading-and-writing-excel-file-in-java/
https://www.geeksforgeeks.org/apache-poi-getting-started/ .
Δοκιμάστε να τρέξετε μερικά παραδειγματάκια μόνα τους για να δείτε πώς δουλεύει η βιβλιοθήκη, πριν μπείτε στη διαδικασία να την εντάξετε στο project σας.

Στοχευμένα κομμάτια κώδικα από το Stackoverflow:

https://stackoverflow.com/questions/72591340/iterate-through-an-array-of-objects-and-write-the-data-into-an-excel-file-in-jav
https://stackoverflow.com/questions/12459181/how-to-add-new-sheets-to-existing-excel-workbook-using-apache-poi
-- με προσοχή βέβαια. Το πιο πιθανό είναι ότι θα χρειαστείτε να χρησιμοποιήσετε στοιχεία όπως

```
import org.apache.poi.ss.usermodel.Cell;
import org.apache.poi.ss.usermodel.CellStyle;
import org.apache.poi.ss.usermodel.Row;
import org.apache.poi.ss.usermodel.Sheet;
import org.apache.poi.ss.usermodel.Workbook;
```

από το ss.usermodel του POI που είναι πιο γενικά και παίζουν και με XSSF και με HSSF (https://poi.apache.org/apidocs/5.0/org/apache/poi/ss/usermodel/package-summary.html).

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 φόρτωσης.
- Ένα πακέτο για την φιλοξενία των domain classes που αφορούν στα tasks.
- Ένα πακέτο για την φιλοξενία των domain classes που αφορούν στα styles.
- Ένα πακέτο για την φιλοξενία των reusable elements, και ιδίως αυτών με τα οποία επικοινωνεί το front-end με το back-end.

Τα παραπάνω είναι ενδεικτικά, βεβαίως, εσείς δεσμεύεστε μόνο από ό,τι σας δίνεται.
Β. Στη front-end πλευρά, σας δίνεται ένα πακέτο για να στεγάσει τη αλληλεπίδραση του συστήματος με το back-end. Εκεί υπάρχει ήδη ένα πακέτο με απλοϊκά console-based παραδείγματα. Αν υλοποιηθεί σωστά το back-end, το εν λόγω front-end παίζει μια χαρά. Μπορείτε να υλοποιήσετε και ένα πιο ωραίο front-end ώστε διαδραστικά να χειρίζεστε Gantt diagrams.

Η κεντρική ιδέα. Σας δίνεται ένα πακέτο app (με υποπακέτα) με τον εξής τρόπο λειτουργίας: ένα σύνολο από naïve client κλάσεις που πρακτικά λειτουργούν εν είδη τεστ, και μια κλάση AppController. 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 naïve client boundary classes και τα οποία ζητά να εξυπηρετηθούν από (β) 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 *utils*, τις οποίες ξέρουν και το front- και το back-end.
Εκτός από τα Interfaces και τις κλάσεις που σας δίνονται έτοιμες, οτιδήποτε άλλο, είναι προς σχεδίαση, υλοποίηση και έλεγχο από εσάς. Αν φτιάξετε και άλλες κλάσεις, π.χ., διαδραστικό console client / GUI / .. front-end, πάλι μέσω του ίδιου AppController έχει νόημα να μιλήσετε με το back-end, χωρίς να τον αλλάξετε (και γι' αυτό υπάρχει άλλωστε).

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).

Υλικό και οδηγίες

Υλικό. Όλο το υποστηρικτικό υλικό για το project θα βρίσκεται στο URL
http://www.cs.uoi.gr/~pvassil/courses/sw_dev/exercises/supportingMaterial/2024-2025/
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) αρχεία με δεδομένα (υπο-φάκελος test/resources) για να κάνετε ελέγχους και για να τρέξετε το σύστημα που θα φτιάξετε, καθώς και το πώς αναμένεται να είναι το περιεχόμενο των reports,
(γ) ένα φάκελο libs με τις σχετικές βιβλιοθήκες.

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:
- Δουλεύουμε υποχρεωτικά με **Eclipse** και την (απλούστερη όλων) **Java 8** για να μπορούμε να διορθώσουμε μαζικά ό,τι υποβάλλετε. Γενικά, ασχέτως μαθήματος, αν δουλεύετε με Java βοηθά να έχετε στημένα τα JDK 8, 11, 17, 21 (δλδ., τις LTS εκδόσεις της Java)
- Θα πρέπει υποχρεωτικά **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_PoiGantt`** (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 & ενδεχομένως να διορθωθεί/αλλάξει/εμπλουτισθεί κάποιο μέρος της εκφώνησης!
- **Πρέπει επίσης να εγγράψετε την ομάδα σας** σε μία φόρμα εγγραφής. Δείτε το web site για τη σχετική ανακοίνωση. Ομάδες που δεν εγγραφούν κινδυνεύουν να μην βαθμολογηθούν.

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 22/12/2024. 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!