 # PROGRESSION

Format : [fait/à faire] TACHE Durée (en minutes)

[v] créer le JOURNAL	5
[v] lire complètement le descriptif général	60
[v] s'inscrire en binôme	2
[v] Makefile	30
[v] Vecteur (+test)	150
[v] Oscillateur (+test)	110
[v] intégrateur d'Euler-Cromer	75
[v] Pendule (+test)	80
[v] Ressort (+test)	30
[v] Systeme (+test)	90
[v] exerciceP10	690
[v] PenduleRessort	70
[v] graphisme: PenduleRessort	110
[v] PenduleRessort	40
[v] graphisme: PenduleDouble	20
[v] espace des phases	130
[v] integrateur Newmark	80
[v] fichier RÉPONSE	15 (en moyenne par semaine)
[v] actualisation des tests	115
[v] fichier CONCEPTION	130
[v] fichier README	50
[v] fichier NOMS	5

==========

 # Semaine 1:
Lecture de la description du projet.
P1: Cette semaine nous avons commencé à écrire la classe vecteur.
Pour le moment, nous avons laissé toutes des méthodes publiques sauf la méthode “essayer_dim”. On a aussi décidé d’ajouter une méthode “getComp” qui retourne la i-ème composante du vecteur et une méthode “size” qui retourne la taille du tableau dynamique qui a la classe Vecteur comme attribut.
C'est possible que pendant les prochaines semaines nous devons changer les droits d'accès aux méthodes.
En ce qui concerne les méthodes “norme” et “norme2”, la valeur de retour pour un vecteur sans dimension est par convention 0.

P3: Nous avons bien réussi à créer le makefile.

Réponse aux questions et mise à jour de journal.

 # Semaine 2 :
Nous avons ajouté les deux constructeurs par défaut demandés. Par contre, nous n’avons pas redéfini le constructeur de copie car nous considérons que celui par défaut nous sert déjà.
En ce qui concerne la surcharge des opérateurs, nous avons implémenté les suivants:
Surcharge Interne

- operator+=
- operator+
- operator-=
- operator-
- operator==
- operator!=
- operator^ (produit vectoriel entre 2 vecteurs de dimension 3)
- operator* (multiplication par un scalaire et produit scalaire entre deux vecteurs)

Surcharge Externe

- operator<<
- operator* (multiplication par un scalaire)

 # Semaine 3 :
On a fini les opérateurs manquants et on a commencé à définir la nouvelle classe Oscillateur du P5.
Mise à jour du fichier réponses.

 # Semaine 4 :

Nous avons ajouté une méthode “FaireUnitaire” à la classe Vecteur. Celle-ci fait unitaire l’instance courante et lance une exception si la norme est égal à 0. Notamment on utilise cette méthode dans le constructeur de la classe Oscillateur pour faire unitaire le vecteur Direction.
Réalisation du test de la classe Oscillateur.
Nous avons implémenté la classe abstraite Intégrateur et sa sous classe IntegrateurEulerCramer.
Réalisation du TestIntegrateur.

 # Semaine 5 :

Implémentation de la classe Ressort et Pendule et réalisation de leurs respectifs tests.
Nous avons aussi modifier la classe Oscillateur, notamment la méthode “eq_mvt” associé à l’équation du mouvement de l’oscillateur en question qui maintenant est virtuelle pure et nous avons aussi enlever un des constructeurs puisque nous considérons ne plus avoir besoin. La méthode “afficher” est aussi devenue virtuelle.
Mise à jour du fichier réponse et du journal.

 # Semaine 6 :

Nous avons fait le tutoriel pour la partie graphique du projet.

 # Semaine 7 :

Implémentation des classes Système, Dessinable et SupportADessin.
Nous avons eu pas mal de soucis avec les dépendances cycliques.
Nous avons décidé d’ajouter une méthode “clone()” au cas où nous aimerions faire des copies polymorphiques.

 # Semaine 8 :

Nous avons raffiné la classe Vecteur, corrigé des méthodes, modifié les droits d’accès, etc.
Nous avons aussi révisé les classes SupportADessin et TextViewer.
Premiers contactes avec QT dans notre projet: affichage des axes.

 # Semaine 9 :

Nous commencé à implémenter la classe RessortPenduleCouplee.
Commencement de l’affichage d’un pendule et un ressort en mode graphique.
Mise à jour du fichier réponses.

 # Semaine 10 :
Finalisation de la classe RessortPenduleCouplee.
Finalisation de l'affichage d'un pendule et un ressort en mode graphique. Nous avons dû ajouter une méthode qui retourne la position de la masse au bout de chaque oscillateur.

 # Semaine 11 :
Implémentation de l'affichage d'un pendule ressort/ ressort pendule en mode graphique.
On a commencé a implémenté l'affichage de la trajectoire dans l'espace des phases.
Finalisation de l'intégrateur de Newmark.
Séparation de fichiers pour les tests.
Mise à jour du fichier réponses.

 # Semaine 12 :
Finalisation affichage de la trajectoire dans l'espace des phases. Ceci a comporté pas mal de changements dans les classes précédentes. Voir fichier CONCEPTION pour plus de détails.
Vérification et mise à jour des tests: Vecteur, Pendule, Oscillateur, Integrateur, Ressort (à cause des nouvelles modifications quelques test n'étaient plus exécutables).
Vérification et mise à jour de l'exercice P9.
Mise à jour de journal et fichier réponses.
Test Intégrateur de Newmark.
Réalisation du fichier NOMS pour le rendu.

 # Semaine 13 :
Modification de la gestion d'exceptions: ajout de QMessageBox pour la simulation en mode graphique.
Décomposition du fichier Integrateur qui jusqu'à maintenant avait les classes Integrateur(abstraite), IntegrateurEulerCramer et IntegrateurNewmark. À présent, nous avons ces classes dans des fichiers séparés.
Implémentation de pendule double.
Préparation pour le rendu du projet:

- Finalisation fichiers CONCEPTION, RÉPONSES, README
- Relecture complète du code et ajout de commentaires
