ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE
EIDGENÖSSISCHE TECHNISCHE HOCHSCHULE – LAUSANNE
POLITECNICO FEDERALE – LOSANNA
SWISS FEDERAL INSTITUTE OF TECHNOLOGY – LAUSANNE

Faculté Informatique et Communication
Cours « Programmation Orientée Objet » (CS-112(g)), SMA+SPH
J.-C. Chappelier

« Programmation Orientée Objet » (SMA+SPH) CS-112(g)

Présentation générale du cours

II MMXX

1 Introduction

Ce document a pour but de vous informer sur la pédagogie du cours « Programmation Orientée Objet » donné aux Sections MA et PH, son mode de fonctionnement et divers autres aspects liés à son organisation.

Veuillez lire l'entièreté de ce document, et ne ratez surtout pas **la section 6** !

Table des matières :

2 Un cours pourquoi ? pour qui ?

L'objectif premier de ce cours est de compléter la formation en programmation commencée au premier semestre afin de vous faire acquérir la base commune nécessaire en programmation pour mener à bien la suite de vos études à l'EPFL. De façon similaire au cours du premier semestre, ses buts incluent :

1. d'enseigner les notions fondamentales communes à la plupart des langages de programmation « orientés objet » ;
2. et de les illustrer au moyen du langage C++.

Les notions vues au semestre d'automne seront consolidées et largement complétées ce semestre, notamment par la réalisation d'un projet. Au terme des deux semestres, chaque étudiant ayant terminé le cours avec succès sera capable de manipuler de programmer de manière indépendante au niveau d'abstration permis par les concepts de base de la programmation orientée-objet.

Ce cours est en premier lieu conçu et organisé comme la suite du cours du premier semester dont les bases doivent donc avoir été acquises. Par ailleurs, bien qu'étant un cours pour programmeurs débutants en programmation orientée objet, le niveau d'exigence restera similaire à celui du premier semestre ; l'ambition étant toujours de faire de vous des programmeurs *compétents*.

Comme au premier semestre, le principe pédagogique fondamental de ce cours est de donner *à chacun* les moyens de progresser à *son* niveau.

Ce principe a conduit à introduire plusieurs éléments :

- un **accès diversifié** au contenu : transparents du cours, exercices, fiches résumé, mini-références, livre conseillé et références externes (« en ligne » et bibliographiques) sont les différents supports mis à disposition ;
- un **accès hiérarchisé par niveau** des élèves : deux niveaux (standard et avancé) pour le contenu du cours (transparents) et quatre niveaux pour les exercices (voir section 3.3.2).

Il est donc primordial que vous identifiez clairement les éléments qui vous sont destinés (débutant ou avancé) :

- pour les débutants : ne vous laissez pas noyer avec des notions trop avancées ;
- pour ceux qui pensent savoir programmer : ne vivez pas trop sur vos acquis et ne vous laissez pas dépasser le moment venu.

Pour cela, différentes indications de niveaux sont données : icône « avancé » dans les transparents, niveaux des exercices, commentaires oraux de l'enseignant, etc. Sachez en faire bon usage !

En plus de vous enseigner le langage C++ lui-même, ce cours a aussi pour objectif de vous sensibiliser à l'importance des aspects *méthodologiques* liés à la programmation. Ainsi, il s'intéresse en priorité au **quoi** (les concepts : encapsulation, abstraction, héritage, polymorphisme, …) et au **pourquoi** (la raison de leur existence). Il va également vous donner des indications sur le **comment** (algorithmique, règles et conseils pour produire de bons programmes, etc.). Mais ce n'est ni un cours d'algorithmique, ni un cours de génie logiciel, seulement un avant-goût de ces disciplines fondamentales pour l'informaticien.

Pour remplir les objectifs multiples de ce cours, les outils pédagogiques suivants sont mis à votre disposition de façon similaire au premier semestre :

- un MOOC (Massive Open Online Course) avec tous ses outils pédagogiques (voir section 3.1) : `https://www.coursera.org/learn/programmation-orientee-objet-cpp/`
- le site du cours sous Moodle :
  `http://moodle.epfl.ch/course/view.php?id=5571`
  sur lequel vous trouverez les transparents du cours, les séries d'exercices et leur corrigé ainsi que des références utiles ;
- des informations générales synthétiques sur différent thèmes de programmation (fiches résumé et mini références ; voir section 5.4) ;
- des forums de discussion : un sur le MOOC, l'autre sur le site Moodle du cours (voir section 5.1).

3 Sous quelle forme le cours est-il donné ?

Le cours est enseigné sous *deux* formes **complémentaires**, aussi importantes l'une que l'autre : MOOC (sur Internet) et présentielle (séances de cours ex cathedra en auditoire et séances d'exercices en salles informatiques).

Les paragraphes suivants expliquent leur rôle respectif et surtout comment organiser votre travail (section 6).

3.1 MOOC (Massive Open Online Course)

Un MOOC de *7 semaines*, « Introduction à la programmation orientée objet (en C++) » [1], a été créé dans le but d'offrir de façon pédagogique, en particulier aux plus débutants, le minimum des bases nécessaires de la programmation orientée objet.

Il constitue à mes yeux la *base principale* de votre apprentissage pour ce cours.

Vous y trouverez :

- des vidéos de cours, d'une dizaine de minutes environ, expliquant en détails un point précis ;
  ces vidéos sont par ailleurs ponctuées de quiz qui vous permettent de vérifier votre compréhension de ce qui est présenté ;
- des quiz hors vidéo dont le but est de vérifier votre acquisition des concepts présentés dans la vidéo ;
- des tutoriels, exercices reprenant des exemples du cours et dont le corrigé est donné progressivement au fur et à mesure de la donnée de l'exercice lui-même ;
  ils sont conseillés comme un premier exercice sur un sujet que l'étudiant ne pense pas encore assez maîtriser pour aborder par lui-même un exercice « classique » ;
- des exercices, libres (le corrigé est donné), vous permettant de **mettre en pratique** les concepts présentés ;
  la pratique *autonome* de ces exercices est une clé fondamentale de votre apprentissage de la programmation ; ne la négligez pas ;
- des « devoirs », exercices à rendre et qui seront corrigés automatiquement.
  Il est important que vous soumettiez ces devoirs notés. Même s'ils n'entrent pas dans la note finale, ils constituent un excellent entraînement pour les examens que vous aurez ici, sur le site de l'Ecole.

3.2 Cours ex cathedra (c.-à-d. en amphi)

Les cours ex cathedra réunissent tous les étudiants dans un auditoire, une heure par semaine.

Durant les deux premiers tiers du semestre (pendant le MOOC), le but de ces cours est de compléter le MOOC en

1. reprenant si nécessaire les *principaux concepts fondamentaux* [2] ;
2. vous montrant, à l'aide d'exemples, comment résoudre des problèmes particuliers (« études de cas ») ;
3. discutant telle ou telle solution ;
4. répondant à vos questions [3].

Pendant le dernier tiers du semestre (après le MOOC), ces cours apporteront des compléments en présentant des thèmes non abordés dans le MOOC.

1. `https://www.coursera.org/learn/programmation-orientee-objet-cpp/`.
2. Il est donc absolument nécessaire que vous ayez vu les vidéos *AVANT* de venir en cours !
3. Il faut donc que vous en ayez !

Afin de faciliter la prise de notes et de vous permettre de préparer vos éventuelles questions, les transparents (ainsi que tout le reste du matériel du cours) seront disponibles sur le site (EPFL) du cours en fin de semaine précédente. Des compléments oraux, sous forme d'exemples additionnels ou de discussions, sont souvent apportés au tableau pendant le cours ex cathedra.

Les examens portant sur la matière qui est enseignée dans les MOOCs **ET** dans les cours ex cathedra, il est important de bien connaître le contenu (et le style d'enseignement) pour arriver à bien se préparer, même si vous ne vous considérez plus comme un novice en programmation.

3.3 Pratique : séances d'exercices et travail à la maison

3.3.1 Organisation générale

Chaque cours ex cathedra est suivi d'une séance d'exercices où les notions théoriques seront mises en pratique. Deux heures d'exercices sont prévues par semaine, mais le temps requis pour résoudre les exercices peut varier, parfois considérablement, en fonction des connaissances préalables et de la préparation de l'étudiant.

Personnellement je conçois qu'un étudiant moyen devrait consacrer deux à trois heures *supplémentaires* de travail personnel par semaine, et un étudiant totalement novice jusqu'à quatre heures. Considérez par ailleurs les deux heures d'exercices affichées à l'emploi du temps comme des heures d'assistance : nous sommes là pour vous aider. Organisez donc votre travail de sorte à faire chez vous les choses qui vous semblent abordables et réservez les aspects difficiles/les questions pour les séances pratiques en salle. Pensez aussi à utiliser les forums (voir plus loin).

Les séances d'exercices sont entièrement pratiques. Elles se déroulent dans une salle d'ordinateurs (voir section 5.2) où vous pouvez travailler seul(e) ou avec des camarades.

Plusieurs assistants (et souvent moi-même) sont présents pendant ces séances pour vous aider. Ils répondront à tous types de questions (sur le cours). Je vous encourage à discuter de vos solutions, vos programmes et de vos problèmes éventuels avec les assistants. Ils ne s'imposent pas, mais attendent que vous les sollicitiez. Sachez profiter pleinement de leur présence !

Pour chaque séance, il y a une série d'exercices à résoudre de manière indépendante. L'énoncé sera mis à disposition sur le site (EPFL) du cours et sur le MOOC en fin de semaine précédente. Le corrigé de la semaine précédente sera également mis à disposition à ce moment là. Tout ce contenu est également disponible dans mon livre « *C++ par la pratique* » (PPUR), en vente à la librairie « La Fontaine », dont une version électronique est également disponible.

Remarque importante : Il est vivement recommandé de participer aux séances d'exercices :
- cela vous permet d'assurer la *régularité* de votre progression qui est une clé essentielle de la réussite au cours ;
- cela vous permet de bénéficier de l'aide des assistants ;
- et cela nous permet de vous donner un retour sur votre niveau afin de vous aider à vous évaluer et si nécessaire à vous indiquer comment progresser.

3.3.2 Catégorisation des exercices par niveaux

Le contenu proposé lors des séries d'exercices est volontairement sur-dimensionné : elles contiennent sensiblement plus de matériel qu'il n'est faisable en deux heures, surtout pour un programmeur débutant ; ceci afin que chacun choisisse selon ce qui l'intéresse, ce qu'il souhaite approfondir.

Il ne vous est donc bien entendu pas demandé de tout faire. Les séries sont à voir comme du matériel d'entraînement dans lequel vous pouvez puiser au gré de vos besoins, un peu comme un livre d'exercices (lequel est par ailleurs vivement recommandé).

Une sélection de ces exercices a été mise sur le site du MOOC.

Dans le but de vous aider à vous organiser, les exercices sont organisés par niveau de difficulté (de 0 à 3) :

- **niveau 0** : reprise pas à pas d'un exemple du cours ; ils peuvent sans problème être sautés par tous ceux qui estiment avoir une suffisamment bonne compréhension de la programmation de base et du cours du jour ; ils doivent par contre, à mon avis, être repris par les novices ;
  ces exercices correspondent aux « Tutoriels » du MOOC ;
- **niveau 1** : ces exercices élémentaires devraient pouvoir être faits par tous dans un temps raisonnable (30 à 45 minutes maximum au début, 20 min. en « régime de croisière », 10 min. max. pour les « pros ») ; ils permettent de travailler les bases ;
- **niveau 2** : ces exercices plus avancés devraient être abordés par tous, sans forcément être finis au début ; la reprise de l'exercice avec la correction devrait constituer un bon moyen de progresser ; c'est par ailleurs le niveau que je considère que vous devrez avoir acquis à la fin du cours ; c'est donc à ce niveau que je fixe le 4.0 des examens ;
- **niveau 3** : ces exercices d'un niveau *avancé* sont pour les plus motivés/habiles d'entre vous ; ils n'existent pas systématiquement dans chaque série ; ils peuvent *dans un premier temps* être ignorés, mais doivent être repris, si nécessaire avec la correction, lors des révisions afin de progresser ;
  les techniques qu'ils utilisent, leur niveau de difficulté, peuvent *ponctuellement* être présents en examen.

Notez que les niveaux sont déterminés en fonction du moment où la série d'exercices est offerte. Il est clair qu'un même exercice donné plus tard au cours de l'année (par exemple au moment de l'examen de fin de semestre) serait considéré comme plus facile !

Je considère que le travail *minimum* par semaine consiste en **deux exercices de niveau 1 et un de niveau 2.**

Bien entendu, votre niveau en programmation ne peut qu'être amélioré si vous réalisez d'avantage d'exercices. Il est donc conseillé, comme déjà évoqué plus haut, de compléter le travail réalisé pendant les séances d'exercices en salle, par du travail personnel hors des heures imparties à ce cours.

Les séries d'exercices hebdomadaires ne sont pas notées (sauf une par semestre, la « série notée », à la 8e semaine) et il ne vous est pas demandé de les rendre. Elles sont à considérer comme une aide à l'apprentissage et comme une préparation aux examens. Si vous n'arrivez pas à terminer une série d'exercices pendant la semaine, il est impératif d'en consulter le corrigé et d'en étudier les détails. Essayez cependant de *résoudre un maximum de problèmes par vous-même*. C'est le meilleur moyen d'apprendre. Si, par contre, il vous reste du temps, complétez la série par un peu de curiosité et d'expérimentation personnelle : ajoutez à votre gré d'autres fonctionnalités à vos programmes, consultez la documentation, etc. En règle générale, toute manipulation sérieuse sur l'ordinateur augmentera vos connaissances.

4 Comment le cours est-il évalué ?

Ce cours est une « branche de semestre » comptant coefficient 4 dans le bloc 2. Ce coefficient est plus élevé (4 au lieu de 3 pour le même plan horaire) afin de prendre en compte la charge de travail supplémentaire (« à la maison ») liée au projet.

Les connaissances que vous aurez acquises seront évaluées à l'aide d'une série notée en 8e semaine, et d'un test écrit la dernière semaine du semestre. De plus, un projet par binôme, réparti sur tout le semestre, viendra compléter l'évaluation [4].

4. Mais la note de projet est majorée par 1.5 fois la moyenne individuelle (sous forme fractionnaire) aux deux autres évaluations ; en clair : on ne peut pas réussir cette branche grâce au projet si par ailleurs on a une moyenne individuelle (hors projet) inférieure à 3.0 (soit une fraction 0.4).

La note finale du cours sera calculée de la façon suivante :

$$\left.\begin{array}{ll} \text{Série notée II} & \text{coef. } 1 \\ \text{Examen théorique II} & \text{coef. } 2 \\ \text{Projet (par binôme)} & \text{coef. } 3\,[4] \end{array}\right\} \text{6 coefficients en tout}$$

Plus précisément : soit p_x le nombre de points obtenus à l'épreuve x (0 en cas d'absence) sur un total maximal pour cette épreuve de t_x ; la note publiée pour cette épreuve est alors l'arrondi suivant :

$$n_x = 1 - 0.25 \left\lfloor -20 \cdot \frac{p_x}{t_x} \right\rfloor$$

et la note finale N est ensuite calculée directement sur les points obtenus (et non pas les notes intermédiaires) par

$$N = 1 - 0.25 \left\lfloor -20 \cdot \frac{\sum_x \theta_x (p_x / t_x)}{\sum_x \theta_x} \right\rfloor$$

où θ_x est le coefficient de l'épreuve x (par exemple 0.33 pour l'examen), et

$$\frac{p_{\text{projet}}}{t_{\text{projet}}} \leq 1.5 \frac{\sum\limits_{x \neq \text{projet}} \theta_x \frac{p_x}{t_x}}{\sum\limits_{x \neq \text{projet}} \theta_x}$$

Je ne demande par contre rien du tout relativement au certificat Coursera : ce certificat est totalement indépendant du présent cours.

5 Supports de cours

5.1 Forums

Il y a plusieurs forums de discussion sur le site du MOOC (Coursera) et sur le site Moodle (EPFL). Ceux sur Moodle (EPFL) sont destinés aux personnes souhaitant poser des questions sur le contenu du cours (au sens large). Il a pour but principal de vous permettre d'interagir avec l'équipe du cours et poser vos questions sans avoir à attendre les séances « officielles ».

Si par contre vous avez des questions relatives au MOOC, à l'apprentissage du C++ en général, aux devoirs notés sur le MOOC (appelés (« *Exercices de programmation* »/« *Programming assignment* » sur la plate-forme Coursera), veuillez alors utiliser les forums du MOOC (Coursera).

Dans tous les cas, **n'hésitez pas à utiliser ces forums** ; ce sont des outils qui peuvent être très riches. Dites-vous bien que si vous vous posez une question, il y a sûrement au moins dix autres de vos camarades (de classe, et 1000 sur le MOOC) qui se posent la même question !

Ainsi, si vous rencontrez des difficultés à résoudre un exercice pendant la semaine, je vous encourage vivement à nous décrire votre problème, si basique soit-il, pour que nous vous aidions à le surmonter. Il n'y a pas de question ridicule, même si certains semblent nettement meilleurs que vous (ce ne sont d'ailleurs pas toujours ceux qui finissent le mieux l'année !). Et tout le monde pourra profiter de la réponse !

Dans tous les cas, **n'**utilisez **pas** les emails personnels pour nous contacter (assistants, enseignant), mais utilisez les forums pour cela. Donc, sauf urgence vraiment personnelle, n'envoyez **aucun message personnel**.

Quelques remarques importantes au sujet des forums :

— Lisez régulièrement les forums car toutes les informations importantes relatives au cours y seront postées : dates, salles et consignes pour les tests, informations sur les cours, notes, changement éventuel au niveau de l'organisation, …
— Le forum ne fonctionne malheureusement habituellement qu'assez tardivement à plein régime ; souvent, par manque de confiance. C'est dommage car il peut vous épargner bien des heures de blocage face à une erreur de programmation.
Je le répète donc, une seule consigne : n'hésitez pas à **y poser vos questions**, et ce dès les premiers cours.

5.2 Ordinateurs

Via 150 « thin clients » (postes de travail) situés dans les salles CO-020, CO-021 et CO-023, ou des connexions à distance depuis votre propre ordinateur (salles CO-120 et CO-121), vous aurez accès chacun à une machine virtuelle Linux (Ubuntu 16.04) tournant sur des serveurs (gros ordinateurs dédiés à cela). Il s'agit des ordinateurs officiels de ce cours (et qui servent aussi à d'autres enseignements).

Les assistants seront à votre disposition dans ces salles pendant les séances d'exercices. Vous pouvez de plus accéder à ces salles quand vous voulez, à condition de ne pas déranger les cours qui s'y déroulent.

Vous pouvez aussi bien travailler sur votre propre ordinateur portable, via une « machine virtuelle » qui sera mise à disposition ou par accès réseau (plus d'explications dans la 1re série d'exercices).

Concernant l'accès aux machines virtuelles, les détails des comptes (ordinateurs et email) vous ont normalement été envoyées suite à votre inscription. Les accès aux salles sont attribués d'office aux étudiants de SMA et SPH. Les autres étudiants (auditeurs libres) devront en faire la demande individuellement (venir me voir). En cas de problèmes d'accès aux salles ou aux ordinateurs, il faut contacter soit le Help-Desk (1234@epfl.ch, tél interne : 1234) si c'est un problème technique, soit le Service Académique (SAC) si c'est un problème administratif, typiquement un problème d'inscription.

Vous comprendrez qu'il est impératif de respecter les directives relatives à l'utilisation des moyens informatiques de l'EPFL. Les jeux sont en particulier interdits dans toutes les salles (sauf ceux que je vous aurais demandé de programmer dans les exercices ☺).

5.3 Transparents, séries et divers supports

La documentation du cours comprend les transparents, les séries d'exercices, quelques livres de référence ainsi que des fiches résumé et des mini-références. Nous utiliserons également la documentation en ligne.

Il n'y a pas de polycopié pour ce cours, mais vous avez accès à toute la documentation qui vient d'être citée soit sur le site lorsque vous êtes à l'EPFL soit par un accès à distance lorsque vous êtes chez vous (voir la rubrique `Conseils et FAQ's` disponible depuis le lien `Références` du site du cours). Les documents et les fichiers seront disponibles en ligne quelques jours avant le cours en question. Vous pourrez ainsi *si nécessaire* en imprimer une version sur les serveurs d'impression de l'Ecole.

Bien que l'ensemble des exercices et corrigés soient accessibles sur le site du cours [5], je vous recommande néanmoins d'acheter le livre d'exercices, non pas parce que cela me donne quelques menus droits d'auteurs, mais surtout parce que cela vous offre un matériel retravaillé, mieux structuré et mieux rédigé, le tout sur un support relié et compact. Enfin, à vous de voir...

A noter également que les PPUR ont publié des « BOOCs », résumés format eBook des vidéos du MOOC :

`https://www.epflpress.org/product/814/9782889143993/introduction-a-la-programmation-orientee-ob`

5. En raison des droits associés à leur publication dans le livre, l'accès Internet à ces exercices et corrigés est interdit hors domaine `.epfl.ch`. En pratique, cela signifie que pour y accéder depuis chez vous, vous devez établir une connexion authentifiée (via VPN).

5.4 Fiches résumé et mini-références

Le but des fiches résumé est double : présenter de façon condensée ce qu'il faut connaître, puis (plus tard pour les programmeurs) avoir un accès très rapide à tel ou tel détail de syntaxe, une fois les concepts connus.

Note : les fiches résumé sont également incluses dans le livre d'exercice, au début de chaque chapitre.

Les mini-références visent par contre à donner une information plus complète sur les aspects techniques du langage C++ et offrent une présentation différente de la matière technique donnée dans le cours. Elles ne sont pas forcément faites pour un programmeur tout à fait débutant mais plutôt pour aller un peu plus loin. À vous de voir, suivant votre niveau, si elles vous conviennent. Mais il n'est pas anormal que certains les trouvent d'un niveau un peu trop avancé.

6 Organisation du travail

6.1 Considérations générales et conseils

J'ai bien conscience que « cette branche n'est qu'une branche pratique », que « vous n'êtes pas en Section Informatique », etc. Et il est bien clair pour moi que le travail doit rester proportionnel à l'importance de la matière pour la filière concernée (c.-à-d. plus concrètement à son coefficient au plan d'études). Mais apprendre la programmation ne peut se faire sans un minimum d'investissement personnel, lequel peut représenter une charge assez lourde, sans être excessive. En organisant correctement leur travail sur l'ensemble de l'année, chaque année un nombre important d'étudiants arrivent très bien à gérer cette charge de travail et réussissent cette branche ainsi que les autres plus importantes pour votre Section.

Le fait que certains étudiants aient la perception de trop travailler dans cette branche, voire que d'autres passent effectivement trop de temps dessus, et pire !, au détriment des autres branches, provient de trois causes principales :

1. une mauvaise gestion des contraintes et des priorités ;
2. une mauvaise répartition du travail dans le temps et au sein du binôme du projet ;
3. une mauvaise évaluation du travail à faire.

Il me parait donc extrêmement important que vous vous fixiez des objectifs (raisonnables) et organisiez correctement votre travail, *dès le début* et tout au long de l'année. N'hésitez pas à me demander conseil dans ce sens si nécessaire ; mais pour résumer :

1. Pour bien apprendre la programmation, il faut *pratiquer*. Cela ne se fait pas du jour au lendemain et demande en effet une certaine quantité de travail *et* une certaine régularité.
2. Néanmoins cette quantité de travail doit rester « raisonnable » (entre 5 et 8 heures grand maximum par semaine *TOUT COMPRIS* (cours + exercices + projet + travail personnel). [6]
3. Il s'agit là d'une branche de semestre pour laquelle vous ne travaillez plus du tout après la fin du semestre. La période sur laquelle vous devez fournir le travail pour cette branche étant plus courte, il est évident qu'à charge totale égale, la charge par semaine devient plus grande.
4. Concernant le projet, celui-ci est également là pour vous apprendre
   (a) à répartir le travail entre vous (en binôme ; ne faites donc pas le travail à double) ;
   (b) à faire ce que vous pouvez en un temps imparti (au lieu de faire le maximum en un temps infini).
5. Fixez-vous un objectif atteignable pour votre niveau : quel est votre objectif ? 6.0 ?

6. Je rappelle que l'EPFL considère que pour un cours de 4 crédits vous devez fournir une charge de travail totale de 8h par semaine : 1 ECTS $\simeq$ 28h de travail sur le semestre (de 14 semaines).

Certains en font beaucoup trop (en particulier sur le projet), principalement en raison de la stimulation des autres groupes (dont certains passionnés) ; la moyenne des projets rendus est souvent très élevée (5 à 5.5). Pour quelqu'un qui a un niveau moyen de 3.5 à 4.5, fournir le travail de développer un projet à un niveau de 5.5 à 6.0 est simplement colossal !

Ce n'est pas ce que *je* vous demande...

(ceci dit, pour obtenir finalement un 4.5, il vaut mieux viser au-dessus (p.ex. 5.0). Faire et rendre un exercice ne signifie pas nécessairement le réussir à 100% et obtenir tous les points).

6. Vous remarquerez que plusieurs semaines de ce semestre n'ont pas de nouvelle donnée pour le projet. Profitez-en pour approfondir votre connaissance des concepts du cours (exercices) et rattraper votre retard dans le projet ; c'est-à-dire à bien vous organiser dans le temps.

6.2 Proposition d'un plan de travail

Afin de bénéficier au mieux de l'outil pédagogique qu'est le MOOC, je vous propose l'organisation suivante :

1. avant le cours en amphi (jeudi 8h15) : regarder la vidéo et faire les quiz (dans et hors vidéo) ;
   temps de travail estimé : entre 1h30 et 2h30 ;
2. jeudi 8h15 : assister au cours, poser des questions
   temps de travail : 45 min. ;
3. avant jeudi 9h15 : finir les quiz (si ce n'est pas fait) et commencer des exercices
   temps de travail estimé : entre 30 min. et 1h30 ;
4. jeudi 9h15–11h00 : séance d'exercices
   temps de travail : 1h45 min. ;
5. entre le jeudi 12h00 et le moment où vous passez à la vidéo de la semaine suivante : faire et rendre le devoir noté du MOOC ; c'est un bon moyen de voir si l'on a compris le sujet ;
   temps de travail estimé : entre 30 min. et 1h30 ;
6. sur le reste de la semaine : avancer sur le projet (sur sa propre partie, ne pas faire le travail à double avec le binôme)
   temps de travail estimé en moyenne par semaine sur le semestre : entre 45 min. et 1h30.

7 Le mot de la fin... ou plutôt du début !

Je le répète donc, une des clés essentielles de la réussite à ce cours est la **régularité** de votre travail et de votre progression. J'ai constaté avec regret lors des années précédentes que certains d'entre vous ne commencent réellement à travailler cette matière qu'au milieu du projet. Il est souvent alors déjà bien trop tard pour combler l'importance des lacunes et les résultats s'en ressentent gravement. N'attendez pas non plus l'approche de la série notée pour nous faire part de vos éventuelles difficultés. Nous sommes à votre écoute *dès le départ* pour vous aider à les surmonter.

Il ne me reste maintenant plus qu'à vous souhaiter un bon apprentissage.

A Annexe : plan du semestre

Voici la vue globale de ce semestre pour le MOOC et les séances présentielles :

		MOOC	déc.	cours 1 h Jeudi 8-9	exercices 2 h Jeudi 9-11
1	20.02.20		0	Intro + compil. séparée	
2	27.02.20	1. Intro POO	0	Intro POO	
3	05.03.20	2. Constructeurs/De	0	Constructeurs	
4	12.03.20	3. Surcharge des op	0	Surcharge	
5	19.03.20	4. Héritage	0	Héritage	
6	26.03.20	5. Polymorphisme	0	Polymorphisme 1	
7	02.04.20	(7. Etude de cas)	1	Polymorphisme 2 / Collections hétérogènes	
8	09.04.20	6. Héritage multiple	1	Héritage multiple	Série notée
-	16.04.20		-	vacances Pâques	
9	23.04.20	(7. Etude de cas)	-	Templates	
10	30.04.20		-	Bibliothèques 1 (SDA)	
11	07.05.20		-	Bibliothèques 2	
12	14.05.20		-	Révisions	
13	21.05.20		-	(Ascension)	
14	28.05.20		-	-	Examen

Le MOOC a été conçu sur 7 semaines pour un travail de 5 à 7 heures par semaines. Pour ce cours ci, j'ai préféré étaler sur deux semaines le sujet délicat du polymorphisme afin de vous offrir plus de pratique et présenter plus de compléments en cours. Cela engendre un léger décalage entre ce cours ci et le calendrier du MOOC à partir de la 7e semaine du cours (semaine 5 du MOOC, lequel commence la deuxième semaine du semestre).