LE GOUVERNEMENT
DU GRAND-DUCHÉ DE LUXEMBOURG
Ministère de l'Éducation nationale,
de l'Enfance et de la Jeunesse

Enseignement secondaire général
Classes supérieures
Division technique générale
Section de l'ingénierie
INFOR / Informatique
Programme
3GIG

Langue véhiculaire :	français
Nombre de leçons :	2
Nombre minimal de devoirs par trimestre :	1 au 1er trimestre , 2 au 2e et 3e trimestre
Dernière mise à jour par la CNES :	31/05/2021

Remarques générales

1. Vu l'aspect pratique du cours, celui-ci doit avoir lieu entièrement dans une salle d'informatique avec un poste par élève. L'utilisation d'une tablette ne se substitue pas à un ordinateur dans ce cadre.
   Le nombre d'élèves par auditoire ou par enseignant doit être limité à 15.
   Il est recommandé d'organiser le cours à raison de blocs de deux leçons consécutives.
2. Ce cours peut être suivi et conclu avec succès uniquement si les élèves peuvent se préparer et s'exercer sur un ordinateur de bureau en dehors des cours. L'accès à un ordinateur qui permet d'installer le JDK Java (donc tournant sous MacOS, Windows ou Linux) est nécessaire.

Initiation à la programmation orientée objet et à l'algorithmique

A. Objectif

La partie II du programme d'informatique de la classe de 3e GIG se comprend comme une initiation à la programmation orientée objet en Java. En ayant conscience des concepts de base de ce type de programmation, les élèves sont capables de résoudre des problèmes simples.

Compétences ciblées	**Contenus**
1. Connaître les notions de base de la programmation orientée objet	- classes - objets - méthodes, paramètres, retour du résultat, le type *void* - attributs - visibilité (*public* et *private*) - le mot clé *new*
2. Savoir modéliser, construire et tester une classe comportant plusieurs méthodes et attributs.	- méthodes (constructeurs, accesseurs, ...) - attributs
3. Connaître et savoir appliquer les opérateurs de base dans des expressions mathématiques et logiques	- variables et types de données simples (int, long, double, boolean, String) - opérateur d'affectation - opérateurs arithmétiques (+, -, *, /, %) - opérateurs de comparaison (==, !=, >, >=, <, <=) - opérateurs logiques (&&, ||, !) - opérateur de concaténation (+) - variables locales - priorité des opérateurs - casting (conversions automatiques et forcées)
4. Connaître et savoir utiliser quelques fonctions de la classe *Math*	- PI - abs, round, random, sqrt, pow
5. Savoir écrire et lire du code source propre et commenté	- indentation du code source - commentaires simples - commentaires JavaDoc (y compris la génération automatique et la lecture des pages générées)
6. Savoir appliquer les structures de contrôle usuelles et savoir résoudre des problèmes qui nécessitent l'utilisation de structures de contrôle imbriquées	- structure alternative if - structure répétitive while avec condition d'entrée (de maintien) - structure répétitive for avec compteur
7. Savoir lire et interpréter des structogrammes	

B. Méthodologie :

Objectifs	Contenus	Remarques
1. Connaître les notions de base de la programmation orientée objet	- classes - objets - attributs - méthodes, paramètres, retour du résultat, le type void - visibilité (public et private) - diagrammes de classes (UML)	- Le mot clé new n'est à traiter qu'à la fin de l'année scolaire (cf. dernier point de cette liste).
2. Introduction au langage Java	- l'architecture d'une classe : déclaration méthodes, constructeurs, accesseurs et attributs - les types de données simples (int, long, double, boolean, String) - les constantes textes	- Le type String est limité à la sortie ou au retour d'un texte. La classe String en tant que telle sera traitée en classe de 12e. - Le mot clé this est à éviter.
3. Savoir appliquer la structure séquentielle dans un algorithme et dans une méthode	- opérateur d'affectation - opérateurs arithmétiques (+, -, *, /, %) - opérateur de concaténation (+) - priorité des opérateurs - casting (conversions automatiques et forcées) - extraits de la classe « Math » - PI - abs, round, random, sqrt, pow - variables locales - indentation du code source - commentaires simples - commentaires JavaDoc (y compris la génération automatique et la lecture des pages générées) - sortie de texte à l'aide de la méthode System.out.print(...), System.out.println(...)	- Cette partie servira surtout à approfondir et à appliquer la matière des deux points précédents. - Dans les exercices, il est recommandé d'employer une logique orientée objet stricte, pour éviter des confusions chez les élèves. (P.ex les calculs devraient s'effectuer de préférence sur des attributs et non des paramètres) - L'introduction de variables locales se fera le plus tard possible pour éviter des confusions avec les paramètres et les attributs.

4. Savoir appliquer la structure alternative dans un algorithme et dans un programme	- structure alternative if complète et réduite - opérateurs de comparaison (==, !=, >, >=, <, <=) - opérateurs logiques (&&, ||, !) - priorité des opérateurs - représentation d'un algorithme sous forme d'un structogramme	- Afin d'assurer une certaine continuité avec le programme d'informatique de la classe de 12e, il va de soi que le concept d'algorithme devra, dès le départ, être explicité à l'aide de structogrammes NSD tels qu'ils sont décrits par la commission nationale des programmes en informatique. - Les élèves doivent être capables d'interpréter des structogrammes et de les traduire en code Java.
5. Savoir appliquer la structure répétitive dans un algorithme et dans un programme	- structure répétitive while avec condition d'entrée (de maintien) - structure répétitive for avec compteur - représentation d'un algorithme sous forme d'un structogramme - algorithmes connus : #1 calcul du PGCD, #2 test si nombre premier	- Il est conseillé de commencer avec une étude approfondie de la structure "tant que" avant d'introduire la structure répétitive commandée par indice "pour". - On insistera sur le problème de la terminaison de l'algorithme contenant la structure répétitive.
6. Savoir combiner et imbriquer les différentes structures de programmation dans un algorithme et dans une méthode	- combinaison et imbrication de structures de contrôle	- Les élèves devront exécuter certains algorithmes sur papier à l'aide d'une grille d'exécution. - Des algorithmes plus complexes sont à développer de préférence d'abord en classe sous forme de structogrammes.
7. Savoir créer des objets sans avoir recours à Unimozer	- le mot clé *new*	

C. Logiciels à utiliser :

Environnement de développement :

- Unimozer (http://unimozer.fisch.lu)

Éditeur de structogrammes :

- Structorizer (http://structorizer.fisch.lu)

D. Liens importants

- Ressources concernant le cours d'introduction à la programmation orientée objet
    - http://java.cnpi.lu/
- JavaDoc
    - https://fr.wikipedia.org/wiki/Javadoc
    - https://www.oracle.com/java/technologies/javase/writing-doc-comments.html

E. Évaluation trimestrielle :

La note trimestrielle se composera :

- des notes obtenues dans les devoirs en classe. La note d'un des devoirs en classe pourra être obtenue par l'évaluation d'un projet informatique.
- d'une note qui évaluera le contrôle continu, c.-à-d. l'effort fourni par l'élève lors des travaux pratiques. Cette note entrera dans la composition de la note trimestrielle pour un tiers au plus.