

Département d'informatique
IFT159 — Analyse et programmation

Plan de cours
Hiver 2014

Enseignant : Benoît Fraikin

Courriel : Benoit.Fraikin@usherbrooke.ca
Local : D4-2009
Téléphone : (819) 821-8000 poste 62013
Site du cours : info.usherbrooke.ca/bfraikin
Disponibilité : lors des laboratoires et sur rendez-vous

Horaire :

Exposé magistral	Mercredi	10h30 à 12h20	Salle D2-1060
	Jeudi	8h30 à 10h20	Salle D2-1060
Exercices/laboratoires	Lundi	16h30 à 17h20	Laboratoire D4-1017

Description officielle de l'activité pédagogique¹

Objectifs	Savoir analyser un problème; avoir un haut degré d'exigence quant à la qualité des programmes; pouvoir développer systématiquement des programmes de bonne qualité, dans le cadre de la programmation procédurale séquentielle.
Contenu	Introduction aux ordinateurs. Analyse et conception de solutions informatiques : simplification, décomposition, modularisation et encapsulation. Critères de qualité : la conformité, la fiabilité et la modifiabilité. Concepts de base de la programmation structurée : séquence, itération, sélection. Modélisation du traitement et modularité : concept de fonctions et d'abstraction procédurale. Concept de base de l'abstraction de données. Introduction aux concepts orientés objet : encapsulation, constructeur, destructeur, surcharge, notation UML. Récursivité. Processus logiciel personnel (PSP0). Revue de code. Tests unitaires.
Crédits	3
Organisation	3 heures d'exposé magistral par semaine 1 heure d'exercices par semaine 5 heures de travail personnel par semaine

Particularités : aucune

¹ <http://www.usherbrooke.ca/fiches-cours/ift159>

1 Présentation

1.1 Mise en contexte

Le cours IFT 159, Analyse et programmation, est le premier cours d'informatique des baccalauréats en imagerie et média numérique, en informatique, en informatique de gestion, en mathématiques et en physique. Il ne présuppose pas de connaissance en programmation. Il ne requiert que la connaissance de l'utilisation usuelle d'un ordinateur. Comme son nom l'indique c'est un cours d'analyse de problèmes et de programmation. L'analyse consiste à l'ensemble des activités dédiées à l'étude détaillée d'un problème. La programmation comprend des activités de conception, de codage, de test et de maintenance de programmes pour ordinateurs. Dans le cadre du cours IFT 159, nous traitons des notions d'analyse, de conception, de codage et de tests. Nous ne parlons pas de la maintenance.

Tout analyste-programmeur qui se voit confier le développement d'un système doit d'abord l'analyser au complet avant de penser à programmer la solution retenue. Cette approche est privilégiée lors des travaux. En effet, pour chaque travail vous devez remettre l'analyse, la conception, le code et les tests reliés à votre solution.

Ce cours est aussi le premier d'une chaîne de cours d'analyse de problèmes et de programmation : en S2 IFT 339, en S3 IFT 232 et IFT 359, puis éventuellement les cours de génie logiciel (IGL). De plus, les connaissances acquises en programmation dans ce cours sont primordiales pour les cours de IFT 209, IFT 215 et IFT 287.

1.2 Objectifs spécifiques

L'objectif du cours est d'apprendre à résoudre un problème en utilisant l'informatique. Le langage utilisé dans le cours est C++. Il est à noter que le but du cours n'est pas d'apprendre un langage mais bien le processus menant à résoudre un problème en utilisant l'informatique.

De façon plus précise, à la fin de cette activité pédagogique, l'étudiante ou l'étudiant sera capable :

1. de comprendre le fonctionnement d'un ordinateur dans le contexte de l'utilisation d'un outil d'élaboration d'une solution programmée ;
2. de lire et comprendre un énoncé de problème peu complexe et procéder à son analyse;
3. d'analyser un problème pour y offrir une solution algorithmique;
4. d'illustrer un algorithme en utilisant la représentation appropriée ;
5. de mettre en œuvre un algorithme à l'aide d'un langage de programmation ;
6. de planifier des tests pour un programme ;
7. de vérifier le bon fonctionnement d'un programme ;
8. de respecter des normes et standards de programmation ;
9. d'utiliser les mécanismes élémentaires d'encapsulation orientée objet;
10. de rédiger un programme en appliquant les principes de base de la dérivation par enrichissements successifs.
11. de documenter la solution au moyen d'un document séparé (analyse et conception) ou d'une documentation incluse (programmation)

1.3 Contenu détaillé

Thème	Contenu	Heures	Objectifs	Travaux
1	Brève introduction aux ordinateurs : modèle pratique du calcul; modèle théorique du calcul; étapes de la mise en œuvre et de l'exécution d'un programme.	3	1	
2	Développement de programmes : cycle de vie et phases de développement du logiciel; éléments de base du C++, notions de programmes (variables, constantes, types, énoncés, fonctions, ...), notions de compilation.	4	2 – 8, 10, 11	A/P
3	Analyse et conception descendante : phases de développement, spécification, analyse, conception, programmation, tests; introduction aux fonctions.	5	2 – 8, 10, 11	A/P
4	Structures sélectives des langages : expressions logiques; énoncé « if », énoncés composés, énoncé « if » emboîté, énoncé « switch »; notions d'analyse et de conception.	5	2 – 8, 10, 11	A/P
5	Structures itératives des langages : concept d'itération, boucle conditionnelle et de comptage; analyse et conception; énoncés « while », « for » et « do..while »; récursivité; boucles emboîtées.	5	2 – 8, 10, 11	A/P
6	Les fonctions et introduction à la récursivité : concept de modularité; fiabilité des fonctions (exactitude, robustesse, assertions); utilisations avancées des fonctions; expressions logiques; récursivité; paramètres de sortie; analyse et conception.	5	2 – 8, 10, 11	A/P
7	Organisation des données et types : représentation interne des données; création de nouveaux types simples; types énumérés; tableaux : concept et utilité des tableaux, tableaux à une dimensions, tableaux à plusieurs dimensions, passage de tableaux en paramètre; types structurés (enregistrements); tableaux d'enregistrements; ensembles; utilisation d'une bibliothèque.	7	2 – 8, 10, 11	A/P
8	Introduction à l'abstraction de données : encapsulation; concepts d'abstraction de données; introduction aux classes; introduction à UML.	8	9	A/P
9	Récursivité : caractéristiques d'un problème récursif; exemples de problèmes récursifs.	3	3 – 5	
10	Introduction à la complexité algorithmique : définition; concept d'ordre de complexité; exemples.	2	3	
11	Conclusion	1		

1. Les heures associées à un thème particulier inclues les heures d'exercices pour un total de 48 heures (douze semaines de quatre heures).
2. Le cours doit comprendre au moins (5) travaux pratiques couvrant tous les sujets marqués dans le tableau. Les lettres « A » et « P » représentent respectivement l'analyse/conception et la programmation.
3. Pour chaque travail, des séries de tests devront être remis ainsi qu'un estimé de l'effort requis pour effectuer le travail (PSPO).

2 Organisation

2.1 Calendrier du cours approximatif

	Période du	Journée de cours	Thème traité	Remarque
1	6 janvier au 10 janvier	lundi 6	—	présentation du plan de cours
		mercredi 8	1	
		jeudi 9	1	TP1A
2	13 janvier au 17 janvier	lundi 13		laboratoire : prise en main
		mercredi 15	2	
		jeudi 16	2	remise TP1A / énoncé TP1P
3	20 janvier au 24 janvier	lundi 20	3	laboratoire
		mercredi 22	—	
		jeudi 23	3	remise TP1P / énoncé TP2A
4	27 janvier au 31 janvier	lundi 27		laboratoire
		mercredi 29	3	activités étudiantes (pas de cours)
		jeudi 30	4	remise TP2A / énoncé TP2P
5	3 février au 7 février	lundi 3	4	laboratoire
		mercredi 5	5	
		jeudi 6	5	remise TP2P / énoncé TP3A
6	10 février au 14 février	lundi 10	4	laboratoire
		mercredi 12	5	
		jeudi 13	5	remise TP3A / énoncé TP3P
7	17 février au 21 février	lundi 17	4	laboratoire
		mercredi 19	5	
		jeudi 20	5	remise TP3P / énoncé TP4A
	22 février au 1 mars	examen de périodique avec levée de cours		
	3 mars au 7 mars	semaine de relâche		
8	10 mars au 14 mars	lundi 10	6	laboratoire
		mercredi 12	6	
		jeudi 13	6	remise TP4A / énoncé TP4P
9	17 mars au 21 mars	lundi 17	—	laboratoire
		mercredi 19	7	
		jeudi 20	7	remise TP4P / énoncé TP5A
10	24 mars au 28 mars	lundi 24	—	laboratoire
		mercredi 26	7	
		jeudi 27	7	remise TP5A / énoncé TP5P
11	31 mars au 4 avril	lundi 31	—	laboratoire
		mercredi 2	8	
		jeudi 3	8	remise TP5P / énoncé TP6
12	7 avril au 10 avril	lundi 7		laboratoire
		mercredi 9	8	
		jeudi 10	8	remise TP6
	11 au 25 avril	à déterminer par la faculté	1 – 10	Examen final récapitulatif

2.2 Méthode pédagogique

Une semaine comprend quatre heures de présence en classe : trois heures de cours dit théorique et une heure d'exercices. Des séances en laboratoire seront interposées afin de permettre la manipulation des concepts vus en cours. Ces séances ne sont pas obligatoires mais peuvent grandement faciliter l'apprentissage des étudiantes et étudiants. La plupart des laboratoires seront libres, c'est-à-dire que la salle sera réservée pour usage du cours mais qu'aucune matière spécifique n'y sera présentée. Une personne du corps enseignant sera disponible pour répondre à des questions durant cette période. Il est cependant important de comprendre que les laboratoires ont pour vocation de placer les étudiantes et étudiants en situation pratique et qu'une partie importante de l'apprentissage en science provient de l'exploration et de la recherche personnelle des moyens de résoudre un problème. Conséquemment, les étudiantes et étudiants devront montrer qu'ils ont déjà bien réfléchi à un problème et ont exploré des voies de solutions avant d'obtenir une réponse de l'enseignant. De plus, tout code présenté à l'enseignant doit être convenablement mis en forme, c'est-à-dire en suivant toutes les indications normatives du département.

Tous les thèmes du cours seront, en général, abordés de la même manière : au moins une étude de cas sera effectuée; le problème sera analysé et une fois correcte, la solution sera programmée; enfin un retour sur les éléments nouveaux du langage vus dans la programmation de la solution sera effectué.

2.3 Évaluation

Devoirs 5 % x 5 : 25 %

Examen périodique : 35 %

Examen final : 40 %

2.4 Échéancier des travaux

TP	Réception du problème à analyser	Remise de l'analyse/conception	Réception de l'analyse/conception à programmer	Remise de la programmation
1	09-01-2014 à 10h30	16-01-2014 à 8h30	16-01-2014 à 10h30	23-01-2014 à 8h30
2	23-01-2014 à 10h30	30-01-2014 à 8h30	30-01-2014 à 10h30	06-02-2014 à 8h30
3	06-02-2014 à 10h30	13-02-2014 à 8h30	13-02-2014 à 10h30	20-02-2014 à 8h30
4	20-02-2014 à 10h30	13-03-2014 à 8h30		20-03-2014 à 8h30
5	20-03-2014 à 10h30	27-03-2014 à 8h30		03-04-2014 à 8h30
6			03-04-2014 à 10h30	10-04-2014 à 23h30

2.5 Directives particulières

L'évaluation se fera par le biais d'un examen intra (35 %), d'un examen final (40 %), de six (6) devoirs dont seuls les cinq (5) meilleurs seront conservés. Chaque devoir compte pour 5 % de la note finale du cours (pour un total de 25 %). Les examens porteront sur toute la matière vue en classe y compris la programmation. Les trois premiers devoirs ont pour but de faire comprendre les principes de base de l'analyse de problème, de l'écriture d'une solution et de la programmation à l'aide du langage vu en cours (le C++). Les trois devoirs suivants ont pour but de vérifier que les étudiantes et les étudiants savent analyser un problème plus complexe, concevoir une solution et finalement implémenter leurs solutions.

Le premier et troisième travaux sont à faire seul. Les autres travaux sont à faire en équipe de deux (2) à trois (3) personnes. Une dispense **exceptionnelle** peut cependant être attribuée si le contexte le demande. La qualité du français ainsi que la présentation pourront être sanctionnée (jusqu'à une pénalité de 5 %). Conformément à l'article 17 du règlement de la Faculté des sciences sur l'évaluation des apprentissages ([UdS2]), l'enseignant peut retourner à l'étudiante ou l'étudiant tout travail non conforme aux exigences quant à la qualité de la langue et aux normes de présentation.

Le non-respect de la date de remise entraînera immédiatement un zéro (0) au devoir, à moins d'un cas **exceptionnel**. Il est à noter qu'un oubli ou un emploi du temps chargé n'est pas un cas exceptionnel. Il en est de même en ce qui concerne une panne d'imprimante ou de votre modem. De plus, le respect des normes départementales est impératif. Un document contenant les normes du département en matière de programmation est fourni sur le site ([IFT159]).

Cette contrainte permet de vérifier que l'étudiant ou l'étudiante sait s'astreindre à une discipline de programmation. Elle permet de plus de mieux insister sur les concepts importants du cours.

La programmation en C++ est réalisée dans l'environnement Unix, qui est la base du système Solaris sur le serveur. Elle peut être réalisée sous tout autre environnement (Windows, Linux ou Mac OS X par exemple), mais devra se trouver et fonctionner sous Unix au moment de la remise du travail. Il est dans votre intérêt d'apprendre à utiliser un minimum le système d'exploitation Solaris pour pouvoir tester vos programmes sur cette plateforme. Un programme qui ne compile pas sur le serveur se verra automatiquement attribuer un zéro (0).

De l'aide technique est disponible sur les ordinateurs des laboratoires, que ceux-ci tournent sous Windows 7, Linux ou Solaris. Il est à noter que l'aide fournie pour travailler hors des laboratoires est limitée par manque de temps et de ressources. Il est conseillé, pour un travail hors des laboratoires, d'utiliser un ordinateur tournant sur Mac OS X, sur Windows XP ou sur Linux avec code: :blocks. Les autres cas de configuration sont peu souhaitables.

2.6 Remarques sur le plagiat

Un document dont le texte et la structure se rapporte à des textes intégraux tirés d'un livre, d'une publication scientifique ou même d'un site Internet, doit être référencé adéquatement. Lors de la correction de tout travail individuel ou de groupe une attention spéciale sera portée au plagiat, défini dans le Règlement des études comme « le fait, dans une activité pédagogique évaluée, de faire passer indûment pour siens des passages ou des idées tirés de l'œuvre d'autrui. ». Le cas échéant, le plagiat est un délit qui contrevient à l'article 8.1.2 du Règlement des études² ([UdS1]) : « tout acte ou manœuvre visant à tromper quant au rendement scolaire ou quant à la réussite d'une exigence relative à une activité pédagogique. » À titre de sanction disciplinaire, les mesures suivantes peuvent être imposées : a) l'obligation de reprendre un travail, un examen ou une activité pédagogique et b) l'attribution de la note E ou de la note 0 pour un travail, un examen ou une activité évaluée. Tout travail suspecté de plagiat sera référé au Secrétaire de la Faculté des sciences.

Ceci n'indique pas que vous n'avez pas le droit de coopérer entre deux équipes tant que la rédaction finale des documents et la création du programme reste le fait de votre équipe. De même, si l'utilisation de morceau de code source ainsi que de documentation provenant du net est autorisée, il ne faut pas que le devoir copie simplement le contenu complet ou partiel du document de référence, mais que l'étudiant ou l'étudiante démontre la compréhension de la partie utilisée. L'enseignant peut, en cas de doute, demander à l'équipe d'expliquer les notions ou le fonctionnement du code qu'il considère comme étant plagié. En cas de doute, ne pas hésiter à demander conseil et assistance à l'enseignant afin d'éviter toute situation délicate par la suite.

² <http://www.usherbrooke.ca/programmes/etude>

3 Documentation et références

- [IFT159] page internet du cours
disponible via info.usherbrooke.ca/bfraikin
contient toutes les informations nécessaires au cours
- [FK04] Frank L. FRIEDMAN et Elliot B. KOFFMAN,
Problem Solving, Abstraction and Design Using C++,
Addison-Wesley. (2004)
- [Ho05] Cay HORSTMAN,
C++ for Everyone,
Pearson / Wiley. (2008)
- [HB08] Cay HORSTMAN et Timothy BUDD,
Big C++ (second Edition),
Pearson / Wiley. (2008)
- [Sa10] Paul SAVITCH,
Absolute C++ (4th Edition),
Prentice Hall. (2010)
- [St09] Bjarne STROUSTRUP,
Programmation : Principes et pratique avec C++,
Pearson. (2009)
- [UdS1] Université de Sherbrooke
Politique d'évaluation des apprentissages
disponible via <http://www.usherbrooke.ca/accueil/fr/direction/documents-officiels/politiques/>
- [UdS2] Université de Sherbrooke
Règlement des études
disponible via <http://www.usherbrooke.ca/accueil/fr/direction/documents-officiels/reglements/>

L'intégrité intellectuelle passe, notamment, par la reconnaissance des sources utilisées. À l'Université de Sherbrooke, on y veille!

Extrait du Règlement des études

8.1.2 Relativement aux activités pédagogiques

L'expression délit désigne d'abord tout acte ou toute manœuvre visant à tromper quant au rendement scolaire ou quant à la réussite d'une exigence relative à une activité pédagogique.

Sans restreindre la portée générale de ce qui précède, est considéré comme un délit :

- a) la substitution de personnes ou l'usurpation d'identité lors d'une activité évaluée ou obligatoire;
- b) le plagiat, soit le fait, dans une activité évaluée, de faire passer indûment pour siens des passages ou des idées tirés de l'œuvre d'autrui;
- c) l'obtention par vol ou par toute autre manœuvre frauduleuse de document ou de matériel, la possession ou l'utilisation de tout matériel non autorisé avant ou pendant un examen ou un travail faisant l'objet d'une évaluation;
- d) le fait de fournir ou d'obtenir toute aide non autorisée, qu'elle soit collective ou individuelle, pour un examen ou un travail faisant l'objet d'une évaluation;
- e) le fait de soumettre, sans autorisation préalable, une même production comme travail à une deuxième activité pédagogique;
- f) la falsification d'un document aux fins d'obtenir une évaluation supérieure dans une activité ou pour l'admission à un programme.

Par plagiat, on entend notamment :

- Copier intégralement une phrase ou un passage d'un livre, d'un article de journal ou de revue, d'une page Web ou de tout autre document en omettant d'en mentionner la source ou de le mettre entre guillemets
- Reproduire des présentations, des dessins, des photographies, des graphiques, des données... sans en préciser la provenance et, dans certains cas, sans en avoir obtenu la permission de reproduire
- Utiliser, en tout ou en partie, du matériel sonore, graphique ou visuel, des pages Internet, du code de programme informatique ou des éléments de logiciel, des données ou résultats d'expérimentation ou toute autre information en provenance d'autrui en le faisant passer pour sien ou sans en citer les sources
- Résumer ou paraphraser l'idée d'un auteur sans en indiquer la source
- Traduire en partie ou en totalité un texte en omettant d'en mentionner la source ou de le mettre entre guillemets
- Utiliser le travail d'un autre et le présenter comme sien (et ce, même si cette personne a donné son accord)
- Acheter un travail sur le Web ou ailleurs et le faire passer pour sien
- Utiliser sans autorisation le même travail pour deux activités différentes (autoplégat)

Autrement dit : mentionnez vos sources.
