UNIVERSITÉ DE SHERBROOKE

## Département d'informatique

# IFT 159 – Analyse et programmation

### Plan de cours
### Automne 2016

## Enseignant


**Vincent Ducharme**

| | |
|---|---|
| Courriel : | Vincent.Ducharme2@usherbrooke.ca |
| Local : | D4-1010-20 |
| Téléphone : | (819) 821-8000 poste 66186 |
| Site : | http://info.usherbrooke.ca/vducharme/ift159 |
| Disponibilité : | Sur rendez-vous |


**Professeur responsable** : Gabriel Girard

## Horaire

| | | | |
|---|---|---|---|
| Exposé magistral : | Lundi | 10h30 à 12h20 | salle D3-2035 |
| | Jeudi | 8h30 à 10h20 | salle D3-2035 |
| Exercices/laboratoires : | Mercredi | 15h30 à 16h20 | salle D4-1023, D4-0023 |

## Description officielle de l'activité pédagogique [1]

| | |
|---|---|
| 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 validité, la fiabilité, la modifiabilité et les tests. 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 : classe, constructeur, surcharge, notation UML (diagramme de classes). Récursivité. Processus logiciel personnel (PSPO). |
| Crédits | 3 |
| Organisation | 3 heures d'exposé magistral par semaine<br>1 heure d'exercices par semaine<br>5 heures de travail personnel par semaine |
| Particularités | Aucune |

1. www.usherbrooke.ca/fiches-cours/ift159

# 1 Présentation

Cette section présente les objectifs spécifiques et le contenu détaillé de l'activité pédagogique. Cette section, non modifiable sans l'approbation d'un comité de programme du Département d'informatique, constitue la version officielle.

## 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 oeuvre 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 :<br>modèle pratique du calcul, modèle théorique du calcul, étapes de la mise en oeuvre et de l'exécution d'un programme. | 3 | 1 | |
| 2 | Développement de programmes :<br>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, notions de gestion d'effort (PSP0). | 4 | 2..8,10,11 | A/P |
| 3 | Analyse et conception descendante :<br>phases de développement, spécification, analyse, conception, programmation, tests fonctionnels; introduction aux fonctions. | 5 | 2..8,10,11 | A/P |
| 4 | Structures sélectives des langages :<br>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 :<br>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 :<br>concept de modularité; fiabilité des fonctions (validité, robustesse, assertions); utilisations avancées des fonctions; expressions logiques; récursivité; paramètres de sortie; tests unitaires et systèmes; analyse et conception. | 5 | 2..8,10,11 | A/P |
| 7 | Organisation des données et types :<br>représentation interne des données; création de nouveaux types simples; types énumérés; tableaux : concept et utilité des tableaux, tableaux à une dimension, tableaux à plusieurs dimensions, passage de tableaux en paramètre; types structurés (enregistrements); tableaux d'enregistrements; ensembles; utilisation d'une bibliothèque («vector»). | 7 | 2..8,10,11 | A/P |
| 8 | Introduction à l'abstraction de données :<br>encapsulation; concepts d'abstraction de données; introduction aux classes; introduction à UML : diagramme de classes. | 8 | 9 | A/P |
| 9 | Récursivité :<br>caractéristiques d'un problème récursif; exemples de problèmes récursifs. | 3 | 3..5 | |
| 10 | Introduction à la complexité algorithmique :<br>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 (12 semaines de quatre heures).
2. Le cours doit comprendre au moins cinq 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 (PSP0).

# 2 Organisation

Cette section propre à l'approche pédagogique de chaque enseignante ou enseignant présente la méthode pédagogique, le calendrier, le barème et la procédure d'évaluation ainsi que l'échéancier des travaux. Cette section doit être cohérente avec le contenu de la section précédente.

## 2.1 Méthode pédagogique

Une semaine comprend quatre heures de présence en classe: trois heures de cours magistral et une heure d'exercices. Une heure est aussi prévues à l'horaire pour de l'assistance en laboratoire. Tous les thèmes du cours, sauf le premier, seront abordés de la même manière : au moins une étude de cas sera étudiée; l'analyse du problème sera effectuée et une fois correcte, la solution sera programmée; enfin on reviendra sur les éléments nouveaux du langage vus dans la programmation de la solution.

Chaque semaine, il y aura environ trois heures d'exposés magistraux décrivant la théorie ainsi que des exemples développés au tableau. Il y aura aussi environ une heure d'exercices qui seront directement intégrés dans les cours magistraux. Au moins une étude de cas sera abordée : analyse de la problématique, implémentation de celle-ci. On reviendra alors sur les éléments nouveaux de langage vus dans la programmation de la solution. Il y a une heure d'assistance en laboratoire prévue à l'horaire. Elle servira principalement à faire du soutien et du dépannage technique. C'est aux étudiants et étudiantes que revient la tâche de s'organiser pour pratiquer ce qui sera vu en cours.

## 2.2 Calendrier

| Semaine | Date | Thèmes | Laboratoire | Lecture |
|---|---|---|---|---|
| 1 | 29 août | Présentation du plan de cours | | |
| 2 | 5 sept. | 1-2 | Présentation de l'environnement | Chapitre 1-2 de [4] |
| 3 | 12 sept. | 2-3 | Environnement et compilation | Chapitre 2-3 de [4] |
| 4 | 19 sept. | 3 | Séquence | Chapitre 3 de [4] |
| 5 | 26 sept. | 4 | Fonctions | Chapitre 4 de [4] |
| 6 | 3 oct. | 4-5 | Sélection/itération | Chapitre 4-5 de [4] |
| 7 | 10 oct. | Examen intra | | |
| 8 | 17 oct. | Relâche | | |
| 9 | 24 oct. | 5 | Sélection/itération | Chapitre 5 de [4] |
| 10 | 31 oct. | 5-6-7 | Tableau | Chapitre 5-6 et sections 9.1, 9.2, 9.4, 9.5 et 11.1 de [4] |
| 11 | 7 nov. | 6 | Organisation et type | Chapitre 6 |
| 12 | 14 nov. | 7 | Type | Sections 9.3, 9.7 et 9.8 de [4] |
| 13 | 21 nov. | 7-8 | Type | Chapitre 10, sections 9.3, 9.7 et 9.8 de [4] |
| 14 | 28 nov. | 8 | Classe | Chapitre 10, sections 9.3, 9.7 et 9.8 de [4] |
| 15 | 5 déc. | 9-10-11 | Au besoin | Chapitre 12, section 11.7 de [4] |
| 16 | 12 déc. - 19 déc. | Révision et Examen final | | |

## 2.3 Évaluation

| Intra | Final | Devoirs | Laboratoires |
|---|---|---|---|
| 25 % | 40 % | 25 % | 10 % |

Conformément à l'article 17 du règlement facultaire d'évaluation des apprentissages [2] l'enseignant peut retourner à l'étudiante ou à l'étudiant tout travail non conforme aux exigences quant à la qualité de la langue et aux normes de présentation.

2. http://www.usherbrooke.ca/sciences/intranet/informations-academiques/reglement-devaluation/

## 2.4 Plagiat

Le plagiat consiste à utiliser des résultats obtenus par d'autres personnes afin de les faire passer pour sien et dans le dessein de tromper l'enseignant. Si une preuve de plagiat est attestée, elle sera traitée en conformité, entre autres, avec l'article 8.1.2 du *Règlement des études* [3] de l'Université de Sherbrooke. L'étudiant ou l'étudiante peut s'exposer à de graves sanctions dont automatiquement un zéro (0) au devoir ou à l'examen en question.

Ceci n'indique pas que vous n'ayez 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. En cas de doute de plagia, l'enseignant peut 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.

## 2.5 Échéancier des travaux

| TP | Réception du problème à analyser | Thème | Remise de l'analyse | Réception de l'analyse à programmer | Thème | Remise de la programmation |
|---|---|---|---|---|---|---|
| 1 | 19/09/16 | Analyse simple | 26/09/16 (à 23h59) | 27/09/16 | Séquence<br>Données simples | 03/10/16 |
| 2 | 03/10/16 | Analyse modulaire | 10/10/16 (à 23h59) | 11/10/16 | Fonctions | 17/10/16 |
| 3 | 17/10/16 | Analyse modulaire | 24/10/16 (à 23h59) | 25/10/16 | Sélection et itération | 31/10/16 |
| 4 | 31/10/16 | Analyse moyenne | 07/11/16 (à 23h59) | | Tableaux 1 dim.<br>Récursivité | 14/11/16 |
| 5 | 14/11/16 | Analyse moyenne | 21/11/16 (à 23h59) | | Données composées<br>Récursivité | 28/11/16 |
| 6 | 28/11/16 | Analyse orientée objet | 05/12/16 (à 23h59) | | Encapsulation | 12/12/16 |

**Directives particulières :**

**Examens**

Les examens porteront sur toute la matière vue en classe y compris la programmation. À moins d'avis contraire, vous aurez droit à deux feuilles de notes recto-verso écrites à la main pour les examens. Aucun ordinateur, téléphone cellulaire, calculatrice ou autre appareils électroniques n'est autorisé lors des examens.

**Devoirs**

1. 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++). Ces premiers travaux sont tous divisés en deux phases. Ils consistent à :
   (a) analyser un problème et
   (b) programmer la solution mise de l'avant d'un autre problème, mais similaire au premier.

   Les 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 leur solution.
2. Les travaux 1 et 3 doivent être effectués individuellement. Les autres travaux se feront obligatoirement en équipe de 2 personnes. Si, pour ces derniers, le travail est effectué seul ou par équipes de trois une pénalité de 50% sera imposée. Cette pénalité sera de 100% pour des équipes de plus que trois.
3. Les remise des devoirs de programmation se feront par la soumission des documents nécessaire grâce à l'outil «`turnin`». Les remises des devoirs d'analyse doivent être remis en version PDF sur `turnin`.
4. Il y a six (6) devoirs dans la session, mais seulement les 5 meilleurs seront comptabilisés. Cependant, le dernier devoir doit obligatoirement être fait et remis. Chaque devoir aura une pondération de 5 points pour un total de 25%.

3. http://www.usherbrooke.ca/programmes/references/reglement/

5. Il y aura de 6 à 10 laboratoires. Cinq de ces laboratoires seront choisis au hasard, corrigés et comptabilisés avec une pondération de 2 points par laboratoire pour un total de 10%.
6. La qualité du français et de la présentation est considérée dans le résultat du travail.
7. Les sujets des travaux seront disponibles sur la page WEB du cours au jour spécifié dans le plan de cours pour la réception du devoir. La remise du travail s'effectue le jour et à l'heure exigés. Le non respect de la date de remise entraîne une pénalité de 25% de la note par jour de retard, 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 électrique, d'ordinateur, d'imprimante, ou du réseau. Cela signifie qu'il faut toujours viser à terminer son travail de programmation au moins 24 heures avant la date de remise pour tenir compte des pannes possibles et de la surcharge quasi-inévitable. Ceci est un conseil qui vaut son pesant de points. Les modalités de remise de chaque travail vous seront fournies avec le sujet de chaque travail.
8. Le respect des normes départementales est impératif. Le document « Norme de documentation des programmes » [2] contient les normes du département en matière de programmation.

   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.
9. Vos programmes doivent pouvoir être compilés autant sous Windows que sous un système basé sur UNIX. Vous ne pouvez pas prendre pour acquis les particularités d'un système d'exploitation, votre code doit donc respecter les standards du langage. Il est quand même dans votre intérêt d'apprendre à utiliser un minimum l'environnement UNIX puisque vous aurez à l'utiliser au cours de votre bac.

   De l'aide technique est disponible sur les ordinateurs des laboratoires, que ceux-ci tournent sous Windows, Linux ou Solaris. Sur le site, les étudiants et étudiantes pourront trouver des liens et de la documentation pour travailler à partir de chez eux. Cependant, aucune autre aide ne sera fournie par manque de temps et de ressources. Il est conseillé, pour un travail hors des laboratoires, d'utiliser l'environnement de développement Code::Blocks qui est disponible sur la plupart des plates-formes.

# 3 Matériel nécessaire pour le cours

Les notes de cours sont disponibles sur le WEB. Le manuel sur lequel est basé le cours est celui de Friedmann [4]. L'achat de ce livre n'est pas obligatoire et est laissé à votre discrétion.

Les normes de programmation du Département d'informatique sont décrites dans [2]. Vous devez absolument vous procurez ce document et le lire.

Même si l'achat du livre n'est pas nécessaire, il est par contre fortement recommandé d'avoir au moins un livre de la liste de référence.

# 4 Références

[1] Yves BOUDREAULT et Wacef GUERFALI : *C++ et un peu + (2e édition)*. Presses Internationales Polytechnique, 2001.

[2] Alex BOULANGER et Félix-Antoine OUELLET : Normes de programmation pour le cours IFT159. http://info.usherbrooke.ca/GabrielGirard/cours/ift-159-analyse-et-programmation/documents/aide-a-la-programmation/normes-de-programmation-ift159, 2014. Aussi disponible sur le site Web du cours.

[3] H. M. DEITEL et P. J. DEITEL : *C++ : How to program*. Prentice-Hall, 2001.

[4] Frank L. FRIEDMAN et Elliot B. KOFFMAN : *Problem Solving, Abstraction, and Design Using C++*. Addison-Wesley, 2004.

[5] Cay HORSTMAN et Timothy BUDD : *Big C++, Second edition*. John Wiley, 2009.

UNIVERSITÉ DE SHERBROOKE

# 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 (autoplagiat)

# Autrement dit : mentionnez vos sources.

UNIVERSITÉ DE SHERBROOKE

## Département d'informatique

# IFT 159 – Analyse et programmation

### Plan de cours
### Automne 2016

## Enseignant


**Vincent Ducharme**

| | |
|---|---|
| Courriel : | Vincent.Ducharme2@usherbrooke.ca |
| Local : | D4-1010-20 |
| Téléphone : | (819) 821-8000 poste 66186 |
| Site : | http://info.usherbrooke.ca/vducharme/ift159 |
| Disponibilité : | Sur rendez-vous |


**Professeur responsable** : Gabriel Girard

## Horaire

| | | | |
|---|---|---|---|
| Exposé magistral : | Mardi | 8h30 à 10h20 | salle D3-2037 |
| | Jeudi | 15h30 à 17h20 | salle D3-2037 |
| Exercices/laboratoires : | Vendredi | 9h30 à 10h20 | salle D4-1017 |

## Description officielle de l'activité pédagogique [1]

| | |
|---|---|
| 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 validité, la fiabilité, la modifiabilité et les tests. 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 : classe, constructeur, surcharge, notation UML (diagramme de classes). Récursivité. Processus logiciel personnel (PSPO). |
| Crédits | 3 |
| Organisation | 3 heures d'exposé magistral par semaine<br>1 heure d'exercices par semaine<br>5 heures de travail personnel par semaine |
| Particularités | Aucune |

1. www.usherbrooke.ca/fiches-cours/ift159

# 1 Présentation

Cette section présente les objectifs spécifiques et le contenu détaillé de l'activité pédagogique. Cette section, non modifiable sans l'approbation d'un comité de programme du Département d'informatique, constitue la version officielle.

## 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 oeuvre 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 :<br>modèle pratique du calcul, modèle théorique du calcul, étapes de la mise en oeuvre et de l'exécution d'un programme. | 3 | 1 | |
| 2 | Développement de programmes :<br>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, notions de gestion d'effort (PSP0). | 4 | 2..8,10,11 | A/P |
| 3 | Analyse et conception descendante :<br>phases de développement, spécification, analyse, conception, programmation, tests fonctionnels; introduction aux fonctions. | 5 | 2..8,10,11 | A/P |
| 4 | Structures sélectives des langages :<br>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 :<br>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 :<br>concept de modularité; fiabilité des fonctions (validité, robustesse, assertions); utilisations avancées des fonctions; expressions logiques; récursivité; paramètres de sortie; tests unitaires et systèmes; analyse et conception. | 5 | 2..8,10,11 | A/P |
| 7 | Organisation des données et types :<br>représentation interne des données; création de nouveaux types simples; types énumérés; tableaux : concept et utilité des tableaux, tableaux à une dimension, tableaux à plusieurs dimensions, passage de tableaux en paramètre; types structurés (enregistrements); tableaux d'enregistrements; ensembles; utilisation d'une bibliothèque («vector»). | 7 | 2..8,10,11 | A/P |
| 8 | Introduction à l'abstraction de données :<br>encapsulation; concepts d'abstraction de données; introduction aux classes; introduction à UML : diagramme de classes. | 8 | 9 | A/P |
| 9 | Récursivité :<br>caractéristiques d'un problème récursif; exemples de problèmes récursifs. | 3 | 3..5 | |
| 10 | Introduction à la complexité algorithmique :<br>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 (12 semaines de quatre heures).
2. Le cours doit comprendre au moins cinq 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 (PSP0).

# 2 Organisation

Cette section propre à l'approche pédagogique de chaque enseignante ou enseignant présente la méthode pédagogique, le calendrier, le barème et la procédure d'évaluation ainsi que l'échéancier des travaux. Cette section doit être cohérente avec le contenu de la section précédente.

## 2.1 Méthode pédagogique

Une semaine comprend quatre heures de présence en classe: trois heures de cours magistral et une heure d'exercices. Une heure est aussi prévues à l'horaire pour de l'assistance en laboratoire. Tous les thèmes du cours, sauf le premier, seront abordés de la même manière : au moins une étude de cas sera étudiée; l'analyse du problème sera effectuée et une fois correcte, la solution sera programmée; enfin on reviendra sur les éléments nouveaux du langage vus dans la programmation de la solution.

Chaque semaine, il y aura environ trois heures d'exposés magistraux décrivant la théorie ainsi que des exemples développés au tableau. Il y aura aussi environ une heure d'exercices qui seront directement intégrés dans les cours magistraux. Au moins une étude de cas sera abordée : analyse de la problématique, implémentation de celle-ci. On reviendra alors sur les éléments nouveaux de langage vus dans la programmation de la solution. Il y a une heure d'assistance en laboratoire prévue à l'horaire. Elle servira principalement à faire du soutien et du dépannage technique. C'est aux étudiants et étudiantes que revient la tâche de s'organiser pour pratiquer ce qui sera vu en cours.

## 2.2 Calendrier

| Semaine | Date | Thèmes | Laboratoire | Lecture |
|---|---|---|---|---|
| 1 | 29 août | Présentation du plan de cours | | |
| 2 | 5 sept. | 1-2 | Présentation de l'environnement | Chapitre 1-2 de [4] |
| 3 | 12 sept. | 2-3 | Environnement et compilation | Chapitre 2-3 de [4] |
| 4 | 19 sept. | 3 | Séquence | Chapitre 3 de [4] |
| 5 | 26 sept. | 4 | Fonctions | Chapitre 4 de [4] |
| 6 | 3 oct. | 4-5 | Sélection/itération | Chapitre 4-5 de [4] |
| 7 | 10 oct. | Examen intra | | |
| 8 | 17 oct. | Relâche | | |
| 9 | 24 oct. | 5 | Sélection/itération | Chapitre 5 de [4] |
| 10 | 31 oct. | 5-6-7 | Tableau | Chapitre 5-6 et sections 9.1, 9.2, 9.4, 9.5 et 11.1 de [4] |
| 11 | 7 nov. | 6 | Organisation et type | Chapitre 6 |
| 12 | 14 nov. | 7 | Type | Sections 9.3, 9.7 et 9.8 de [4] |
| 13 | 21 nov. | 7-8 | Type | Chapitre 10, sections 9.3, 9.7 et 9.8 de [4] |
| 14 | 28 nov. | 8 | Classe | Chapitre 10, sections 9.3, 9.7 et 9.8 de [4] |
| 15 | 5 déc. | 9-10-11 | Au besoin | Chapitre 12, section 11.7 de [4] |
| 16 | 12 déc. - 19 déc. | Révision et Examen final | | |

## 2.3 Évaluation

| Intra | Final | Devoirs | Laboratoires |
|---|---|---|---|
| 25 % | 40 % | 25 % | 10 % |

Conformément à l'article 17 du règlement facultaire d'évaluation des apprentissages [2] l'enseignant peut retourner à l'étudiante ou à l'étudiant tout travail non conforme aux exigences quant à la qualité de la langue et aux normes de présentation.

2. http://www.usherbrooke.ca/sciences/intranet/informations-academiques/reglement-devaluation/

## 2.4 Plagiat

Le plagiat consiste à utiliser des résultats obtenus par d'autres personnes afin de les faire passer pour sien et dans le dessein de tromper l'enseignant. Si une preuve de plagiat est attestée, elle sera traitée en conformité, entre autres, avec l'article 8.1.2 du *Règlement des études* [3] de l'Université de Sherbrooke. L'étudiant ou l'étudiante peut s'exposer à de graves sanctions dont automatiquement un zéro (0) au devoir ou à l'examen en question.

Ceci n'indique pas que vous n'ayez 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. En cas de doute de plagia, l'enseignant peut 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.

## 2.5 Échéancier des travaux

| TP | Réception du problème à analyser | Thème | Remise de l'analyse | Réception de l'analyse à programmer | Thème | Remise de la programmation |
|---|---|---|---|---|---|---|
| 1 | 19/09/16 | Analyse simple | 26/09/16 (à 23h59) | 27/09/16 | Séquence<br>Données simples | 03/10/16 |
| 2 | 03/10/16 | Analyse modulaire | 10/10/16 (à 23h59) | 11/10/16 | Fonctions | 17/10/16 |
| 3 | 17/10/16 | Analyse modulaire | 24/10/16 (à 23h59) | 25/10/16 | Sélection et itération | 31/10/16 |
| 4 | 31/10/16 | Analyse moyenne | 07/11/16 (à 23h59) | | Tableaux 1 dim.<br>Récursivité | 14/11/16 |
| 5 | 14/11/16 | Analyse moyenne | 21/11/16 (à 23h59) | | Données composées<br>Récursivité | 28/11/16 |
| 6 | 28/11/16 | Analyse orientée objet | 05/12/16 (à 23h59) | | Encapsulation | 12/12/16 |

**Directives particulières :**

**Examens**

Les examens porteront sur toute la matière vue en classe y compris la programmation. À moins d'avis contraire, vous aurez droit à deux feuilles de notes recto-verso écrites à la main pour les examens. Aucun ordinateur, téléphone cellulaire, calculatrice ou autre appareils électroniques n'est autorisé lors des examens.

**Devoirs**

1. 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++). Ces premiers travaux sont tous divisés en deux phases. Ils consistent à :
   (a) analyser un problème et
   (b) programmer la solution mise de l'avant d'un autre problème, mais similaire au premier.

   Les 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 leur solution.
2. Les travaux 1 et 3 doivent être effectués individuellement. Les autres travaux se feront obligatoirement en équipe de 2 personnes. Si, pour ces derniers, le travail est effectué seul ou par équipes de trois une pénalité de 50% sera imposée. Cette pénalité sera de 100% pour des équipes de plus que trois.
3. Les remise des devoirs de programmation se feront par la soumission des documents nécessaire grâce à l'outil «`turnin`». Les remises des devoirs d'analyse doivent être remis en version PDF sur `turnin`.
4. Il y a six (6) devoirs dans la session, mais seulement les 5 meilleurs seront comptabilisés. Cependant, le dernier devoir doit obligatoirement être fait et remis. Chaque devoir aura une pondération de 5 points pour un total de 25%.

3. http://www.usherbrooke.ca/programmes/references/reglement/

5. Il y aura de 6 à 10 laboratoires. Cinq de ces laboratoires seront choisis au hasard, corrigés et comptabilisés avec une pondération de 2 points par laboratoire pour un total de 10%.
6. La qualité du français et de la présentation est considérée dans le résultat du travail.
7. Les sujets des travaux seront disponibles sur la page WEB du cours au jour spécifié dans le plan de cours pour la réception du devoir. La remise du travail s'effectue le jour et à l'heure exigés. Le non respect de la date de remise entraîne une pénalité de 25% de la note par jour de retard, 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 électrique, d'ordinateur, d'imprimante, ou du réseau. Cela signifie qu'il faut toujours viser à terminer son travail de programmation au moins 24 heures avant la date de remise pour tenir compte des pannes possibles et de la surcharge quasi-inévitable. Ceci est un conseil qui vaut son pesant de points. Les modalités de remise de chaque travail vous seront fournies avec le sujet de chaque travail.
8. Le respect des normes départementales est impératif. Le document « Norme de documentation des programmes » [2] contient les normes du département en matière de programmation.

   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.
9. Vos programmes doivent pouvoir être compilés autant sous Windows que sous un système basé sur UNIX. Vous ne pouvez pas prendre pour acquis les particularités d'un système d'exploitation, votre code doit donc respecter les standards du langage. Il est quand même dans votre intérêt d'apprendre à utiliser un minimum l'environnement UNIX puisque vous aurez à l'utiliser au cours de votre bac.

   De l'aide technique est disponible sur les ordinateurs des laboratoires, que ceux-ci tournent sous Windows, Linux ou Solaris. Sur le site, les étudiants et étudiantes pourront trouver des liens et de la documentation pour travailler à partir de chez eux. Cependant, aucune autre aide ne sera fournie par manque de temps et de ressources. Il est conseillé, pour un travail hors des laboratoires, d'utiliser l'environnement de développement Code::Blocks qui est disponible sur la plupart des plates-formes.

# 3 Matériel nécessaire pour le cours

Les notes de cours sont disponibles sur le WEB. Le manuel sur lequel est basé le cours est celui de Friedmann [4]. L'achat de ce livre n'est pas obligatoire et est laissé à votre discrétion.

Les normes de programmation du Département d'informatique sont décrites dans [2]. Vous devez absolument vous procurez ce document et le lire.

Même si l'achat du livre n'est pas nécessaire, il est par contre fortement recommandé d'avoir au moins un livre de la liste de référence.

# 4 Références

[1] Yves BOUDREAULT et Wacef GUERFALI : *C++ et un peu + (2e édition)*. Presses Internationales Polytechnique, 2001.

[2] Alex BOULANGER et Félix-Antoine OUELLET : Normes de programmation pour le cours IFT159. http://info.usherbrooke.ca/GabrielGirard/cours/ift-159-analyse-et-programmation/documents/aide-a-la-programmation/normes-de-programmation-ift159, 2014. Aussi disponible sur le site Web du cours.

[3] H. M. DEITEL et P. J. DEITEL : *C++ : How to program*. Prentice-Hall, 2001.

[4] Frank L. FRIEDMAN et Elliot B. KOFFMAN : *Problem Solving, Abstraction, and Design Using C++*. Addison-Wesley, 2004.

[5] Cay HORSTMAN et Timothy BUDD : *Big C++, Second edition*. John Wiley, 2009.

UNIVERSITÉ DE SHERBROOKE

# 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 (autoplagiat)

# Autrement dit : mentionnez vos sources.