Construction et maintenance de logiciels

Groupe 20
Mardi, de 17h30 à 21h00 PK-1320 (cours)

Jeudi, de 17h30 à 20h00 PK-S1540 et PK-S1555 (atelier)

Responsable(s) du cours

Nom du coordonnateur : SALAH, Aziz

Nom de l'enseignant : CHIEZE, Emmanuel
Local : PK-4115
Téléphone : (514) 987-3000 #3699
Courriel : chieze.emmanuel@uqam.ca
Site Web : aucun

Description du cours

Initier les étudiants à la programmation à l'aide d'un langage impératif et procédural.
Familiariser les étudiants à la construction professionnelle de logiciels et à leur maintenance.

- Notions de base de la programmation procédurale et impérative en langage C sous environnement Unix/Linux (définition et déclaration, portée et durée de vie, fichier d'interface, stuctures de contrôle, unités de programme et passage des paramètres, macros, compilation conditionnelle).
- Décomposition en modules et caractéristiques facilitant les modifications (cohésion et couplage, encapsulation et dissimulation de l'information, décomposition fonctionnelle).
- Style de programmation (conventions, documentation interne, gabarits).
- Déboggage de programmes (erreurs typiques, traces, outils, par ex., gdb).
- Assertions et conception par contrats.
- Tests (unitaires, intégration, d'acceptation, boîte noire vs. boîte blanche, mesures de couverture, outils d'exécution automatique des tests, par exemple, xUnit, scripts).
- Évaluation et amélioration des performances (profils d'exécution, améliorations asymptotiques vs. optimisations, outils).
- Techniques et outils de base pour la gestion de la configuration (par exemple, make, cvs).
- Introduction à la maintenance de logiciels (types de maintenance, techniques de base, par exemple, remodelage, automatisation des tests de régression).

Préalables académiques :

Ce cours comporte une séance obligatoire de laboratoire (2 heures).

Objectifs du cours

Ce cours vise à introduire les étudiants à la construction professionnelle de logiciels. Il vise aussi à familiariser les étudiants avec la programmation procédurale et impérative.

À la fin du cours, l'étudiant devrait être capable :

- de développer et modifier des composants logiciels écrits dans un langage impératif et procédural;
- de bien maitriser le langage C et le compilateur du C sous UNIX/Linux;
- d'utiliser les notions de module, de cohésion et couplage, de complexité structurale, de dissimulation de l'information, etc., pour évaluer la qualité d'un composant logiciel;
- d'expliquer et d'utiliser les principales techniques de modularisation : décomposition fonctionnelle, dissimulation de l'information, filtres et pipelines;
- d'utiliser des assertions (pré/post-conditions, invariants) pour documenter des composants logiciels et assurer leur bon fonctionnement;
- de déboguer un programme à l'aide de techniques, stratégies et outils appropriés;
- de vérifier le bon fonctionnement d'un composant logiciel à l'aide de tests fonctionnels et structurels, et d'évaluer à l'aide des notions et outils appropriés la qualité des tests (par ex., complexité cyclomatique, mesures de couvertures des tests);
- d'évaluer de façon empirique à l'aide d'outils appropriés (par ex., profils d'exécution) les performances d'un composant logiciel de façon à pouvoir, si nécessaire, en améliorer les performances;
- d'utiliser divers outils (outil de gestion de configuration, fichiers makefile, langage de scripts) pour organiser le développement de programmes comportant plusieurs composants ou modules;
- d'expliquer les notions de base de la maintenance des logiciels et d'appliquer certaines techniques de maintenance (tests de régression exécutés automatiquement, remodelage de programmes).

Contenu du cours

Introduction générale au Langage C

- Types de base
- Conversions, expressions et priorités des opérateurs
- Entrées/sorties
- Instructions de contrôle
- Tableaux
- Fonction main

Les pointeurs en C

- Fonctions avec passage de référence
- Liens entre pointeurs et tableaux
- Chaines de caractères
- Tableaux à deux dimensions
- Arithmétique des pointeurs, pointeurs de ligne
- Fonctions avec tableau/chaine de caractères
- Allocation dynamique
- Arguments de programme

Les structures et les unions en C

- Manipulations de base des structures
- Les tableaux de structures
- Les structures et les pointeurs
- Structures auto-référencées
- Structures et fonctions
- Champs de bits
- Unions

Les fichiers en C

- Opération de bases : fopen, fscanf, fprinft, fclose, feof, fseek, etc
- Fichiers binaires
- Canaux standards : stdin, stdout, stderr

Le préprocesseur C et macros

- Directive #include
- Les macros
- L'inclusion conditionnelles

Les modules en C

- Porté, visibilité et linkage d'une variable
- Variable globale, variable statique
- Structure d'un fichier entête .h
- Exemple d'un programme avec modules
- Makefile

La conception

- Objectif de la conception
- Approches de décomposition :
    - dirigée par le contrôle
    - dirigée par les données
    - dirigée par les responsabilités
- Abstraction par paramétrisation, par spécification, des données(ADT)
- Modules et leurs qualités :
    - Six niveaux de cohésion,
    - Cinq niveaux de couplage

Gestion de configuration et autres

- git (cvs, subversion ou équivalent, couvert au labo)
- gdb (couvert au labo)
- Documentation et style (article)
- Évaluation et amélioration des performances (profils d'exécution, améliorations asymptotiques vs. optimisations, outil gprof au labo)

Test de logiciel

- Test boite blanche :
    - métriques de couverture
    - graphe de flux de control
    - tests des chemins de base
    - tests des boucles
- Test boite noire :
    - test à base de modèle (automate, graphe)
    - test en boite noire basé sur les partitions d'équivalence
    - cas d'analyse des valeurs limites
- Définition des types de tests :
    - test unitaire
    - test de sous système
    - test système
    - test d'intégration
    - test d'acceptation
    - test de régression
- Test unitaire
    - environnement de test
    - exemples d'application avec les tests en boite noire et test en boite blanche
- Outils de tests (au labo)
    - CUnit au labo
    - shell script : bash
    - autres outils Unix (diff, sed, grep, awk etc)
- Test d'intégration

    - intégration descendante
    - intégration ascendante
- Procédure de débogage, bonnes pratiques, types de maintenance, remodelage

Programmation défensive

- Définition : erreur, faute, défaillance
- Principes de la programmation défensive
- Assertions
- Traitement des erreurs

Modalités d'évaluation

Description sommaire	Date	Pondération
Examen Intra		30 %
Examen Final		30 %
TP1		15 %
TP2		25 %

Les examens sont individuels et à livres ouverts (documentation papier seulement, aucune documentation électronique). Une moyenne d'au moins 50 % aux examens est exigée pour réussir le cours.
L'étudiant ne remettant aucun des travaux se verra attribuer un échec.
Les travaux pratiques peuvent être réalisés individuellement ou en équipe de deux (2) personnes. Les retards dans la remise des travaux ne sont pas acceptés, sauf en cas d'entente PRÉALABLE avec l'enseignant. La qualité du français sera prise en considération (jusqu'à 10 % de pénalité).

Les règlements concernant le plagiat seront strictement appliqués. Pour plus de renseignements, consultez le site suivant :
http://www.sciences.uqam.ca/etudiants/integrite-academique.html

Politique d'absence aux examens

L'autorisation de reprendre un examen en cas d'absence est de caractère exceptionnel. Pour obtenir un tel privilège, l'étudiant-e doit avoir des motifs sérieux et bien justifiés.

Il est de la responsabilité de l'étudiant-e de ne pas s'inscrire à des cours qui sont en conflit d'horaire, tant en ce qui concerne les séances de cours ou d'exercices que les examens. **De tels conflits d'horaire ne constituent pas un motif justifiant une demande d'examen de reprise.**

Dans le cas d'une absence pour raison médicale, l'étudiant-e doit joindre un certificat médical original et signé par le médecin décrivant la raison de l'absence à l'examen. Les dates d'invalidité doivent être clairement indiquées sur le certificat. Une vérification de la validité du certificat pourrait être faite. Dans le cas d'une absence pour une raison non médicale, l'étudiant-e doit fournir les documents originaux expliquant et justifiant l'absence à l'examen – par exemple, lettre de la Cour en cas de participation à un jury, copie du certificat de décès en cas de décès d'un proche, etc. Toute demande incomplète sera refusée. Si la direction du programme d'études de l'étudiant-e constate qu'un étudiant a un comportement récurrent d'absence aux examens, l'étudiant-e peut se voir refuser une reprise d'examen.

L'étudiant-e absent-e lors d'un examen doit, dans les cinq (5) jours ouvrables suivant la date de l'examen, présenter une demande de reprise en utilisant le formulaire prévu, disponible sur le site Web du département à l'adresse suivante : **http://info.uqam.ca/politiques/**

L'étudiant-e doit déposer le formulaire dûment complété au secrétariat de la direction de son programme d'études : PK-3150 pour les programmes de premier cycle, PK-4150 pour les programmes de cycles supérieurs. Pour plus de détails sur la politique d'absence aux examens du Département d'informatique, consultez le site web suivant : **http://info.uqam.ca/politiques**

Intégrité académique

PLAGIAT Règlement no 18 sur les infractions de nature académique. (extraits)

Tout acte de plagiat, fraude, copiage, tricherie ou falsification de document commis par une étudiante, un étudiant, de même que toute participation à ces actes ou tentative de les commettre, à l'occasion d'un examen ou d'un travail faisant l'objet d'une évaluation ou dans toute autre circonstance, constituent une infraction au sens de ce règlement.

La liste non limitative des infractions est définie comme suit :

- **la substitution de personnes;**
- **l'utilisation totale ou partielle du texte d'autrui en la faisant passer pour sien ou sans indication de référence;**
- **la transmission d'un travail pour fins d'évaluatiion alors qu'il constitue essentiellement un travail qui a déjà été transmis pour fins d'évaluation académique à l'Université ou dans une autre institution d'enseignement, sauf avec l'accord préalable de l'enseignante, l'enseignant;**
- **l'obtention par vol, manoeuvre ou corruption de questions ou de réponses d'examen ou de tout autre document ou matériel non autorisés, ou encore d'une évaluation non méritée;**
- **la possession ou l'utilisation, avant ou pendant un examen, de tout document non autorisé;**
- **l'utilisation pendant un examen de la copie d'examen d'une autre personne;**
- **l'obtention de toute aide non autorisée, qu'elle soit collective ou individuelle;**
- **la falsification d'un document, notamment d'un document transmis par l'Université ou d'un document de l'Université transmis ou non à une tierce persone, quelles que aoient les circonstances;**
- **la falsification de données de recherche dans un travail, notamment une thèse, un mémoire, un mémoire-création, un rapport de stage ou un rapport de recherche;**
- **Les sanctions reliées à ces infrations sont précisées à l'article 3 du Règlement no 18.**

Les règlements concernant le plagiat seront strictement appliqués. Pour plus de renseignements, veuillez consulter les sites suivants : **http://www.sciences.uqam.ca/etudiants/integrite-academique.html** et **http://www.bibliotheques.uqam.ca/recherche/plagiat/index.html**

Médiagraphie

VC Blaquelaire, J.P. -- Méthodologie de la programmation en C (4e édition) -- Dunod, 2005. [QA76.73 C15B75].
VC Stephen Prata, C Primer Plus (5th Edition), Sams Publishing, 2004.
VC Roger Pressman, Software engineering : A practitioner’s approach, 7 ed., McGraw-Hill, Inc., New York, NY, USA, 2010.
VC Ian Sommerville, Software engineering, 9 ed., Addison Wesley Longman Publishing, Redwood City, CA, USA, March 13, 2010.
VC Barbara Liskov and John V. Guttag, Program development in java - abstraction, specification, and object-oriented design., Addison-Wesley, 2001.
VC Steve Maguire, Writing solid code : Microsoft’s techniques for developing bug-free C programs, Microsoft Press, Bellevue, WA, USA, 1993.
Boris Beizer, Software testing techniques (2nd ed.), Van Nostrand Reinhold Co., New York, NY, USA, 1990.
VC Kernighan, B.W. et Ritchie, D.M. -- Le langage C -- deuxième édition, Masson, Paris, 1990.
VC Kernighan, B.W. et R. Pike, R. -- The Practice of Programming -- Addison-Wesley, 1999.

VC Kernighan, B.W. et R. Pike, R. -- La programmation - En pratique -- Vuibert, 2001. [QA76.6K48814].

VC Loukides, M. et Oram, A. -- Programming with GNU Software -- O'Reilly, 1997.

VC McConnell, S. -- Code Complete - A Practical Handbook of Software Construction -- (Second edition). Microsoft Press, Redmond, WA, 2004.

VC Purdy, G.N. -- CVS Précis & concis -- Éditions O'Reilly, 2004.

VC ZELLER, A. and KRINKE, J. -- Essential Open Source Toolset -- John Wiley & Sons, Ltd, 2005.

UR http://www.info2.uqam.ca/~tremblay/INF3135
Site web contenant des notes de cours et transparents, des anciens examens, etc.

A : article - C : comptes rendus - L : logiciel	C : complémentaire - O : Obligatoire - R : recommandé
S: Standard - U : uri - V : volume	