ICT Programming Curriculum Meeting

December 3 2015

Agenda with Meeting Minutes

1) Meeting Notes – October 8 2015
2) Subject Outlines Review
   a. IPC144-BTP100
   b. OOP244
   c. OOP345-BTP305
   d. DSA555-BTP500
   e. Action - Revisions
3) Fall 2015 Review
   a. Web site changes
   b. Mid-semester statistics
   c. Assessment weights going forward
   d. Directions for the winter semester
4) Discussions
   a. Announcements
   b. Programming topics - break, continue, goto, etc.
   c. Compilers - Visual Studio IDE, gcc -Wall
   d. Workshop scripts
   e. Weekly meetings
   f. Material allowed on tests
5) Proposals
   a. Augmenting 1st semester delivery
   b. C++ best practices (Idioms)
   c. Mapping topics to core literacies
   d. When/how to teach/learn the tools in current fashion
6) Other Business

Attendees: Patrick Crawford, Fardad Soleimanloo, Catherine Leung, Edgardo Arvelaez, Elnaz Delpisheh, Eden Burton, Mark Fernandes, Marek Laskowski, Arta Kogan, Ian Tipson, Mary Lynn Manton, Greg Blair, Rahul Thomas, Hasan Kamal-al-Deen, Joseph Hughes, Chris Szalwinski

Regrets: Elliott Coleshill, Peter McIntyre, Tim McKenna, Mariam Daoud

Agenda Approved by Attendees

Meeting Notes – October 8 2015

http://zenit.senecac.on.ca/wiki/imgs/Curriculum_Meeting_October_8_2015.pdf

Meeting Notes for Oct 8 will be approved unless objections submitted before December 15 2015.

No objections received meeting notes deemed approved

Subject Outlines Review

IPC144 – Introduction to Programming Using C

Learning Outcomes (to be approved by T&L)

1. Design functions using selection and iteration constructs to solve a programming task
2. Connect functions using pass-by-value and pass-by-address semantics to assemble a complete program
3. Design collections using arrays and structures to manage data efficiently
4. Code algorithms using standard library functions to incorporate existing technology
5. Stream data using standard library functions to interact with users and access persistent text
6. Trace the execution of a procedural program to validate its correctness
7. Code complete programs using appropriate object and pointer types to solve programming problems
8. Explain the purposes of procedural programming features to inform business persons

BTP100 – Programming Fundamentals Using C

Learning Outcomes (to be approved by T&L)

1 - Design functions using selection and iteration constructs to solve a programming task
2 - Connect functions using pass-by-value and pass-by-address semantics to assemble a complete program
3 - Design collections using arrays and structures to manage data efficiently
4 - Code algorithms using standard library functions to incorporate existing technology
5 - Stream data using standard library functions to interact with users and access persistent text
6 - Trace the execution of a procedural program to validate its correctness
7 - Develop algorithms using procedural programming concepts to communicate coding plans
8 - Code complete programs using appropriate object and pointer types to implement specified coding plans
9 - Explain the purposes of procedural programming features to inform business persons

OOP244 – Introduction to Object Oriented Programming

Learning Outcomes (to be approved by T&L)

1. Design classes with dynamically allocated resources to model the components of a programming solution
2. Design member functions using logic constructs to solve tasks of linear complexity
3. Relate classes using inheritance hierarchies to minimize the duplication of object code
4. Design polymorphic objects to amplify the reusability of program code
5. Use stream objects to interact with users and access persistent data
6. Trace the execution of object-oriented code to validate its correctness
7. Code a complete program using polymorphic objects to solve a systems or business problem
8. Explain the purpose of an object-oriented programming feature to inform a business person

BTP200 – The Object Oriented Paradigm Using C++

Learning Outcomes (to be approved by T&L)

1 - Design classes with dynamically allocated resources to model the components of a programming solution
2 - Design member functions using logic constructs to solve tasks of linear complexity
3 - Relate classes using inheritance hierarchies to minimize the duplication of object code
4 - Design polymorphic objects to amplify the reusability of program code
5 - Use stream objects to interact with users and access persistent data
6 - Trace the execution of object-oriented code to validate its correctness
7 - Code a complete program using polymorphic objects to implement a specified coding plan
8 - Explain the purpose of an object-oriented programming feature to inform a business person
9 - Develop an algorithm using object-oriented concepts to solve a simple programming problem

OOP345 – Object Oriented Software Development Using C++

Learning Outcomes (approved by T&L)

1 - Design collections of model objects using sequential containers and raw and smart pointers to manage a programming solution
2 - Create function objects and closures to customize an algorithm for a particular task
3 - Model polymorphic behavior using inheritance, interfaces, virtual functions and templates to minimize code duplication
4 - Implement algorithms using the standard template library to solve a systems or business problem
5 - Design multi-tasked solutions using threading libraries to improve a program's performance
6 - Design file stream objects to backup text and binary data for future restoration

BTP305 – Object Oriented Software Development Using C++

Learning Outcomes (approved by T&L)

1 - Design collections of model objects using sequential containers and raw and smart pointers to manage a programming solution
2 - Create function objects and closures to customize an algorithm for a particular task
3 - Model polymorphic behavior using inheritance, interfaces, virtual functions and templates to minimize code duplication
4 - Implement algorithms using the standard template library to solve a systems or business problem
5 - Create algorithms of quadratic complexity to solve non-linear problems
6 - Design multi-tasked solutions using threading libraries to improve a program's performance
7 - Design file stream objects to backup text and binary data for future restoration
8 - Trace the execution of program code that includes a linked list to debug an application

DSA555 – Data Structures and Algorithms

Still to be done

- Formulate a comprehensive set of learning outcomes

Learning Outcomes – will need to update to meet Seneca QA Standards

- ~~En~~Incorporate data structures into the applications they write
- Implement searching and sorting algorithms
- Create their own data structures
- Determine which algorithm or data structure to use in different situations

BTP500 – Data Structures and Algorithms

Still to be done

- Formulate a comprehensive set of learning outcomes

Learning Outcomes – will need to update to meet Seneca QA Standards

- methodically test and debug complex Java and C++ programs
- analyze a programming problem in order to determine the suitability of implementing a variation on a well-known data structure or algorithm as part of the solution
- compare the relative efficiencies of competing algorithms
- determine the shortcomings or limitations of a particular data structure or algorithm
- use the data structures provided in the standard Java and C++ libraries
- describe the features and limitations of a particular data structure or algorithm
- implement variations of well-known data structures and algorithms using Java and C++
- identify, through the application of research techniques, and use in a programming solution, well known data structures or algorithms not specifically covered in class
- produce technical and usage documentation for data structures/algorithms developed

Actions – Including Revisions to Subject Outlines

IPC144 and BTP100

- Outlines need to be tweaked to conform to College Quality standards
  - Decrease the number of outcomes (five or six is ideal)
  - Increase the distinction between degree and diploma courses
  - Need comments and feedback before submission to T&L
- We need to add testing and debugging to the learning outcomes
  - Testing and debugging should be embedded into the teaching itself
  - We don't necessarily need a separate workshop on testing and debugging

OOP244 and BTP200

- Outlines need to be tweaked to conform to College Quality standards
  - Decrease the number of outcomes (five or six is ideal)
  - Increase the distinction between degree and diploma courses
  - Need comments and feedback before submission to T&L

Topic List, Evaluations, Reference Material

Participating faculty initiative -> Coordinator approval -> Posting

⇨ Peter – deadline for faculty recommendations was Nov 26 2015

Timeline, Workshop Learning Outcomes

Participating faculty initiative (week 0) -> Posting

⇨ Workshop LOs to be decided by multi-section faculty by week 0 – all of us need to recognize the flow of content through each course and with respect to the set of core courses – workshop LOs are not open to revision by individual faculty, who are expected to abide by the initial multi-section decision (CS)

⇨ Changes to the timeline or LOs of workshops in any multi-section course have to be discussed with all instructors teaching the course

⇨ Add a separate section for reflection at the end of each workshop. Reflections allow the students to feedback what they have learned or not learned. This will also give the instructors better sense of the students' understanding and performance

⇨ Timeline Posted Start of Week 0 (Jan 8 2016)
  - Part of contract with the students – hence: Mandatory
  - Individual faculty to communicate variations to students caused by scheduling and holiday conflicts

IPC144 and BTP100

- At the end of each workshop create a number of test cases – the student must write down the correct output and what is produced by their program
- Testing and debugging should be included some or all workshops
- Add testing and debugging to the learning outcomes of relevant workshops
- **Marek:** To change/reduce the number of workshops from 10 to 9 – keep the materials covered and LOs but reduce the number of workshops, perhaps mixing two workshops or

breaking a workshop and spreading it across several others – the main concern is to keep the number of topics and LOs.
  - **One good suggestion:** part of the lab must be submitted in class

OOP244 and BTP200

  - **Fardad:** To change/reduce the number of workshops – same comments apply as for IPC144
  - **Lab assistant request:** Fardad suggested adding 2 hours to lab assistants so that they can give feedback to students. Or perhaps the learning centre should give feedback. **Mary Lynn** will think about this – it may impinge on broader issues. (to be continued).

DSA555 and BTP500

- **Cathy:** to work on the subject outlines

Fall 2015 Mid-Semester Review

Web Site Modifications

IPC144 - Web Site Dynamic Component (Workshops Assignments, Instructors)

- Redirection to the current Semester's Content
- https://scs.senecac.on.ca/~ipc144/dynamic/workshops/index.html

BTP100 - Web Site Dynamic Component (Workshops Assignments, Instructors)

- Redirection to the current Semester's Content
- https://scs.senecac.on.ca/~btp100/dynamic/workshops/index.html

OOP244 - Web Site Dynamic Component (Workshops, Final Project, Instructors)

- Redirection to the current Semester's Content
- https://scs.senecac.on.ca/~oop244/dynamic/workshops/index.html

BTP200 - Web Site Dynamic Component (Workshops, Final Project, Instructors)

- Redirection to the current Semester's Content
- https://scs.senecac.on.ca/~btp200/dynamic/workshops/index.html

Clarification

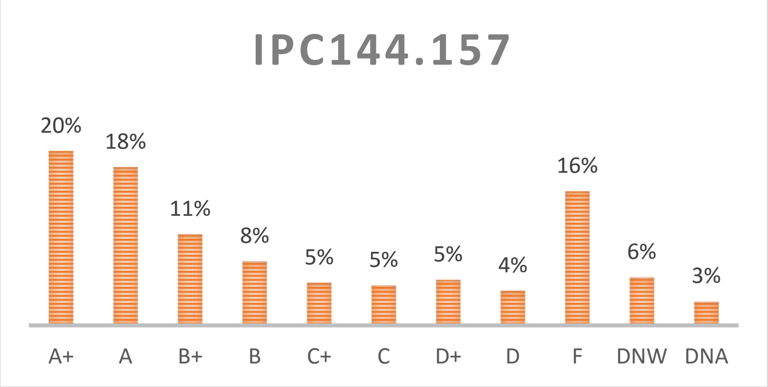
- IPC144, OOP244, BTP100 and BTP200 – notes and resources are uncoupled from the workshops and assignments:
  - Notes and resources are under ../pages and backed-up to ../pages.??? each semester
  - Workshops and Assignments are under ../dynamic to facilitate changes each semester
  - We need multiple sets of workshops developed by instructors teaching these courses to build a repository

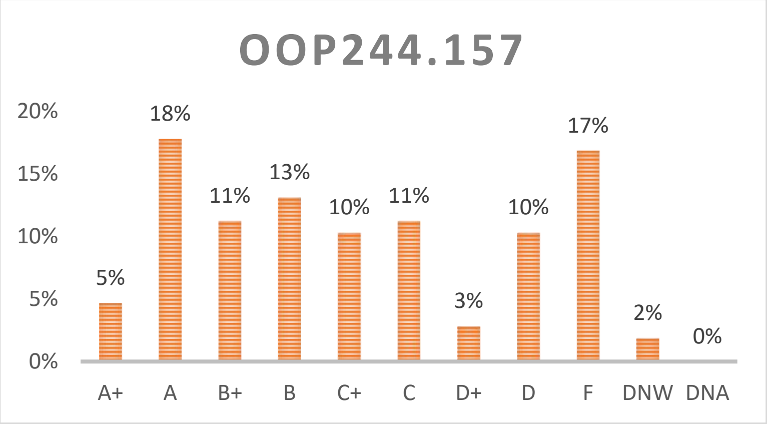
OOP345 BTP305 - Revised Timeline

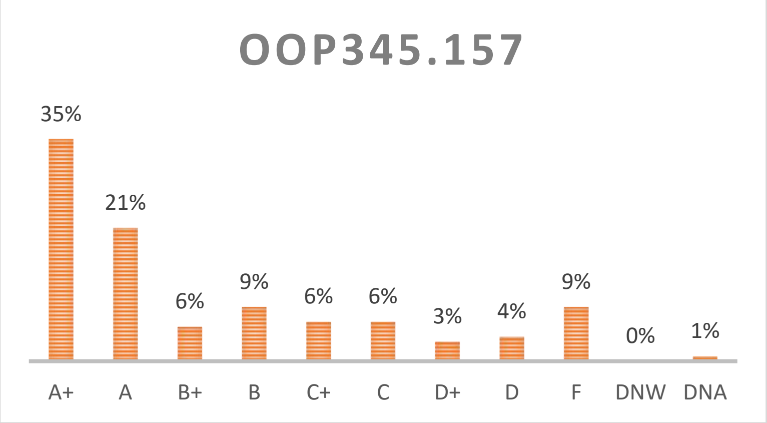
- Direct path to STL with detail features deferred to last third of the semester
- https://scs.senecac.on.ca/~oop345/pages/timeline.html
- https://scs.senecac.on.ca/~btp305/pages/timeline.html

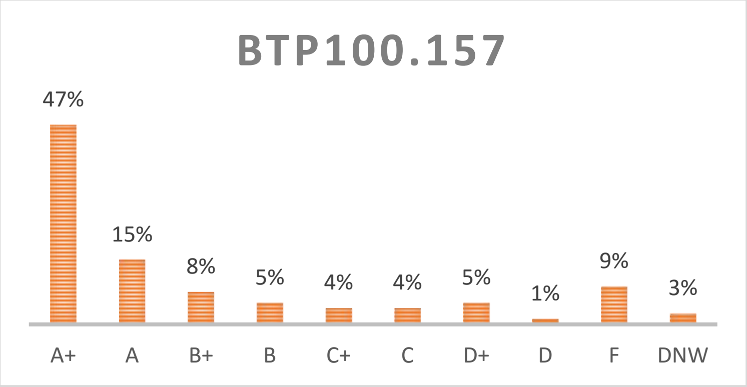
Mid-Semester Statistics

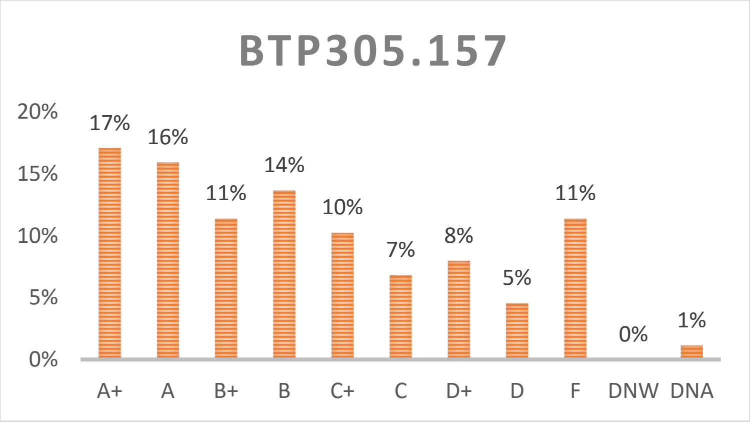
- Per Instructor distributions are available on request











Evaluation Weights for the Winter Semester

	IPC144	OOP244	OOP345	BTP100	BTP200
Final Project	20%	20%	20%	20%	20%
Workshops/Lab Work	30%	30%	30%	30%	30%
Quizzes	15%	15%	15%	15%	15%
Midterm Test	20%	20%	15%	20%	20%
Final Exam	15%	15%	20%	15%	15%

Discussions

Announcements

CERN Document Server   Travel Software Using GPU Hardware

Programming Topics

Break, continue and goto

- Deferred to next meeting

Compilers

Windows and Linux Environments

- Do we make Visual Studio IDE a standard tool in this course?
  - Fardad, Elliott use Visual Studion
  - Mark, Greg use Ubuntu
- Is matrix Compilation and Execution sufficient?

gcc –Wall

- We need to discuss environments for coding
- Codeblocks
- Codelight
- **Peter McIntyre:** Visual Studio may be too much for a student who's new to programming. We can configure its startup environment to be clutter-free, enabling easy onboarding, while enjoying some of its goodness (code sense, coloring, and so on). We just need to discuss this widely before deciding, and then proceed carefully.

Workshop Scripts

Experience with Interactive Scripts for Workshops and the Final Project

- Deferred to next meeting

Weekly Meetings

Collaboration between Diploma and Degree Sections

- Mentioned but discussion deferred to next meeting

Material Allowed on Tests and the Final Exam

Do we restrict the material allowed to what is stated on the subject outline?

Tim McKenna: YES: During IPC144 exam 2015.4 (Summer), there were 4 academic honesty warnings issued as a result of “any C programming textbook” [Any C programming textbook AND a single reference sheet (double sided) is permitted and must be submitted with the exam]. Textbooks and a single reference sheet result in a lot of “stuff” on desks during an exam and may increase opportunities for cheating.

Peter McIntyre: Allowing a book and a reference sheet suggests to me that maybe the question format is a problem. Frequent use of a book and/or reference sheet implies to me that answering a question is simply a lookup exercise. If a student has good lookup skills, then we award grades. Is that really what we’re testing? Alternatively in a question that asks them to write code, a few things comes to mind. The code shoud be shor, and ensure that the topic’s principles are understood. We should be relaxed about precise syntax – do we really care about a missing brace or semi-colon? We shouldn’t be asking for lengthy code answers. If we’re testing things like function definition and calling (in IPC144), then I would argue that NO reference should be permitted, as that is basic knowledge. Moving on to higher-level courses, class definition, its members for construction and destruction, etc. should also be basic knowledge. My bias (for years) is no references permitted.

Proposed Winter Working Groups

- Volunteers for All Groups Needed

Augmenting 1st Semester Delivery

- Deferred to next meeting

C++ Best Practices (Idioms)

C++ does not protect the software developer from misuse of the language. Idioms are examples usage that has been popularized by the experts. Applying these idioms assists in producing quality code.

Understanding idioms helps an instructor explain why C++ code is written in a certain way rather than in any arbitrary way.

https://en.wikibooks.org/wiki/More_C%2B%2B_Idioms (June 18 2014)

"C++ has indeed become too "expert friendly" at a time where the degree of effective formal education of the average software developer has declined. However, the solution is not to dumb down the programming languages but to use a variety of programming languages and educate more experts. There has to be languages for those experts to use–and C++ is one of those languages." Bjarne Stroustrup (2006). "The Problem with Programming" Computer News Nov 28 2006.

https://www.gitbook.com/book/lefticus/cpp-best-practices/details

https://isocpp.org/wiki/faq/Coding-standards

http://www.amazon.com/exec/obidos/ASIN/0321113586/

http://stroustrup.com/JSF-AV-rules.pdf

- Deferred to next meeting

Mapping Topics to Core Literacies

- Deferred to next meeting

How/When to Teach/Learn the Tools in Current Fashion

- Deferred to next meeting

Other Business

- Meeting adjourned 17:00