COSC 102 Spring 2020

INTRO TO COMPUTING II

Course Description

This course teaches advanced programming, including abstract data types (ADTs), algorithms that manipulate them and analysis of algorithms. Information hiding, data abstraction, and modular design are emphasized. They are exemplified by studying important data structures (DS), such as lists, stacks, queues, trees and hashmaps, and their algorithms that maintain sequences or search effectively. Object-Oriented Programming (OOP) in Java is the first learning unit, and is reinforced by the following units.

In projects students design and implement programs that illustrate the topics of the course. By the end of the course, students should be able to design and implement multi-module programs.

Goals

- Study several important data structures and different implementation techniques
- Learn how to choose the "best" data structures for a specific context
- Learn how to modify standard data structures for specific purposes, and create new ones
- Learn to read, understand, interpret and extend programs written by others in the same language
- Understand and recognize proper programming style and make design decisions consistent with OOP practice
- Learn to reuse classes, provided code and libraries and the standard Java library, which is the Application Programming Interfaces (APIs)
- Learn to write clean OOP programs: correct, well-designed and easy to read code

Prerequisite: COSC 101 or equivalent

This course does not require knowledge of Python, the programming language used in COSC 101. It does, however, assume knowledge of programming fundamentals in an imperative programming language, including the following concepts.

- Primitive and reference variables (including lists or arrays and strings).
- Looping (definite and indefinite).
- Conditionals.
- Functions and pass-by-value semantics for arguments.

Meeting times

Section	Instructor	Room	Time
COSC 102 A	Fourquet E.	MCGREG 315	MWF 9:20–10:10

Instructors

Name	Email	Phone	Office	Office Hours
Fourquet E.	efourquet	x 6033	MCGREG 309	M 11:30–12:30 W 12:30–2:00 F 10:30–11:30

Name	Email	Phone	Office	Office Hours
Lyboult M.	mlyboult	x 7564	MCGREG 318	M 9:00–11:00 T 1:30–3:30 F 8:45–10:10 F 1:00–2:00 Recitation

Open Lab

COSC tutors are available in MCGREG 329 during open lab hours. While lab tutors can help with labs and projects, the most important is for you to understand the course materials and topics. Get to know the lab tutors and learn from them. Your goal is to become independent! Open lab does not exist after 102...

Matt's Recitation

On Friday afternoons from 1:00PM–2:00PM Matt will hold a weekly *"Recitation"* in MCGREG 329. This time will be devoted to a high-level recap of that week's topics, summarizing what you learned in lectures and readings. Matt will also take questions from students on particular concepts or materials anyone would like to review in more detail. Matt will send out an email a day or two beforehand outlining the planned topics. Some weeks may be devoted solely to helping students with a particular assignment or project. These hours are meant to be relatively casual, so feel free to drop by/leave whenever you like!

Materials

Required Textbook *Building Java Programs: A Back to Basics Approach* **BJP**

by Stuart Reges and Marty Stepp. **Fourth Edition** available at the bookstore (3rd Edition available on the web).

Websites Students are responsible for keeping up-to-date with content of the followings websites.

Course Webpage https://www.cs.colgate.edu/~efourquet/cosc102/index.html

for course general information: lecture topics and readings, labs and projects.

Moodle http://moodle.colgate.edu

for homework submissions and possible course section announcements, discussion and extra materials.

Software

All programming is done using Java. Classroom and lab computers have Java 7 and jEdit installed. Links under the Resources section on course webpage provide instructions for installation on personal computers.

Live Coding CodingBat Java

CodingBat is a free website with live coding problems to build coding skill. The coding problems give immediate feedback, providing an opportunity to practice and solidify understanding of the concepts. By doing CodingBat problems students put reading concepts into practice.

Course Work

This course and its associated lab count for a total of 1.25 credits. Therefore, students are expected to spend about **12.5 hours/week** studying for this course.

Attendance It is **mandatory** to come to class. In class we cover concepts and discuss technical, social and ethical topics and practice that cannot be learned on your own from a textbook. Thus, I reserve the right to **lower by one grade** your final performance in the case you have missed **more than 4 classes**. In addition, students are responsible for material covered, distributed and completed in class and in lab.

Participation As learning cannot be passive, I expect active participation of **all** the members of the class. **Courteous and shared** participation is demanded, in lectures, labs and office hours.

In class students are expected to comment, to ask and answer questions about readings and lecture content, and to collaborate with their peers on problem solving exercises. Students are expected to show a level of involvement that enhances everyone's learning. Participation also refers to **listening to others**, meaning both the professor and classmates. Encouraging your peers to be involved in our reflective exchanges enhances everyone learning. No-one should dominate the conversation; we should all be respectful of our shared space and everyone's voices.

In lab there will be tutors assisting the lab instructor. I encourage you to ask questions on concepts and processes. (Asking how to get the exercice done doesn't help you to learn.) You will often work collaboratively and be taught how to do so: it is an important competence to develop. Please pay attention and respect our instructions on the process to work well in team.

Office hours are a special time for me to teach you one-on-one. I encourage you to attend my office hours. I will teach you **how to work independently**. I will ask you questions in response to your question, to help you think further. My role is to help you develop competences and understanding. It is on purpose I don't give answers, I model with you the process of solving problems. I will help you on a first step and often ask you to continue close by so I can help another student and will return to you thereafter.

Arriving late, stepping out of the classroom, using cell phone or not paying attention disturb everyone participation. Disrespect of our classroom will **negatively** affect your final grade beyond the participation component.

Readings/Exercises Readings are assigned for each lecture. You are required to complete them **prior** coming to the class. Lectures and discussions do not duplicate assigned material. You should focus on the beginning of the reading, learn to read diagrams, and take notes during your reading. Anything from the required readings, even if not directly discussed in class, is fair game for the homework, exams and reading quizzes. Reading quizzes—many opened-notes—will take the **first ten minutes of class** so you need to *arrive on time* to successfully give your answers. I will not give you extra time to make up for your late arrival.

Projects You will usually be working on a project. Each project takes the concepts covered in lectures and practiced in lab and pushes students to apply them to challenging context. Because projects are substantial and long they will have a milestone due date, sometimes requiring a demo in lab, which will be part of the project grade.

Exams There will be four exams: three exams during the term and one final, in addition to the reading quizzes. All exams are to completed individually; absolutely no collaboration is allowed. *Exam dates will be posted on the course website.* Notify the instructor at least *one week prior to the exam* about scheduling conflicts.

Lab To complete this course, students must also sign up for a 2-hour weekly laboratory section. Labs are **not always** designed to be completed during the lab period. *The lab is a separate course with a separate grade.* Students should come to the lab prepared by having done the required readings and practice exercises.

Grading

The final grade for the class is calculated on the following weighting. Grading is on an absolute scale (i.e., no curve). Note that the lab grade is separate and will be provided in the lab syllabus.

Coursework	Percent
Reading quizzes, homeworks & hand-outs	10
Interim Exams (3 x 8)	24
Final exam	24
Projects	36
Participation	6

To pass the course, you must pass the final exam. If your projects grade is more than one letter higher than the rest of the coursework, the final grade may be adjusted downward. Interim exams will also count towards your lab grade.

Final course grades are determined as follows. As a general rule, fractions are rounded down (e.g., an 89.9 is a B+, not an A-). A grade of A+ is awarded only when a student demonstrates truly exceptional performance and is not simply determined by having a high final course grade.

A+	A	A-	B+	B	B-	C+	C	C-	D+	D	D-	F
*	≥ 93	90-92	87-89	83-86	80-82	77-79	73-76	70-72	67-69	63-66	60-62	< 60

Policies

Academic honesty and collaboration You are expected to abide by Colgate's Code of Student Conduct and by Colgate's Academic Honor Code.

Collaboration (i.e., discussing the problem and possible solutions) while working on lab or project is fine, but the work you submit must be your own. Roughly speaking, it is okay to share ideas but it is not okay to share any artifacts (code, write-up, etc.). Here is a good way to think about it: you and a classmate can get together, discuss ideas, and even write some code. However, you are expected to leave that meeting with *nothing* – no notes and certainly no code – and write up *your own solution.* If someone helped you or you collaborated with peer(s) state it clearly with any submitted work: write down their names in the main header file or better in a `readme` file. *Failing to acknowledge your collaborators can be considered a violation of the honor code.*

Late homework or missing work Late projects are not generally accepted. Start project early and submit early and often. Missed work may receive a zero. Conflicts with in-class exams should be addressed well in advance.

Unexpected circumstances If unexpected circumstances arise that might compromise your performance in the course (inability to attend class, complete the homework on time, etc.), please let me know as soon as possible so that we may arrange appropriate accommodations. Usually these accommodations will be made in consultation with your administrative dean.

Getting Help

A key to your success at Colgate, and in life, is figuring out what resources are available and using them to help you achieve your goals. For any homework problems or other class-related questions that you have, there are several options for getting help. Please take advantage of these opportunities!

1. See instructor during office hours.
2. Post a question on the Moodle forum.
3. Form a study group with other students in the class and work together on a regular basis (note the collaboration policy above).
4. See CS student tutors during Open Lab hours.
5. Send us email.

In addition, please be aware of the great resources that Colgate provides.

CS laptop The department has a limited number of computers available for temporary loan. You must request and obtain permission to borrow equipment. Taking a laptop from the classroom without signing an official form is not permitted. If you are interested in borrowing a computer, please come talk to me.

Tech Support for Students The ITS Desk offers peer support and expertise related to computers and technology. Located in Case-Geyer the team assists with problems concerning email, internet, and public access computers on campus.

Academic Support and Disabilities Services If you feel you may need an accommodation based on the impact of a disability, you should contact your instructor privately to discuss your specific needs. If you have not already done so, please contact Lynn Waldman, Director of Academic Support and Disability Services at 315-228-7375 in the Center for Learning, Teaching, and Research. Ms. Waldman is responsible for determining reasonable and appropriate accommodations for students with disabilities on a case-by-case basis, and more generally, for ensuring that members of the community with disabilities have access to Colgate's programs and services. She also assists students in identifying and managing the factors that may interfere with learning and in developing strategies to enhance learning.

Counseling Center College life can sometimes get bumpy; if you are experiencing emotional or personal difficulties, the Counseling Center offers completely confidential and highly professional services. Their staff are trained to help students manage a wide array of emotions. The Counseling Center meets with over half the student body for clinical services at some point during their four years at Colgate. You can arrange an appointment online or by phone (315-228-7385). For emergencies, a counselor is available 24/7 by calling campus safety at 315-228-7333 and asking for the counselor on call.

Student Health Services The services provide high-quality, accessible, convenient, cost-effective, non-judgmental, and confidential health care for all students. You can arrange an appointment at the main clinic (next to Community Memorial Hospital) by phone (315-228-7750), or visit the satellite walk-in clinic (lower level of Curtis Hall) during normal business hours for assessment of minor injuries and illnesses.

Haven is a sexual violence response center that provides confidential care, support, advocacy, and trauma-informed clinical services for survivors of sexual assault, intimate partner violence, child/family abuse, stalking, and/or harassment. Haven works from a survivor-centric model that is intended to honor and empower individuals to define their experiences and healing process. Haven's role is to provide options and facilitate finding support. You can call (315-228-7385) or visit (Garden Level of Curtis Hall) during business hours. After hours, call campus safety at 315-228-7333 and ask for the counselor on call; you can also go directly to the Community Memorial Hospital Emergency Room, and the attending nurse will contact an advocate from Help Restore Hope Center to discuss your options.

Topics

- ...
- [OOP] Object-oriented Programming (3 weeks)
- [BA] Basic Analysis (1 week)
- Programming Practices

Object-oriented Programming

Concepts

- Object-oriented design
  - Decomposition into objects carrying state and having behavior
  - Class-hierarchy design for modeling
- Definition of classes: fields, methods, and constructors
- Subclasses, inheritance, and method overriding
- Dynamic dispatch: definition of method-call
- Subtyping
  - Subtype polymorphism; implicit upcasts in typed languages
  - Notion of behavioral replacement: subtypes acting like supertypes
  - Relationship between subtyping and inheritance
- Object-oriented idioms for encapsulation
  - Privacy and visibility of class members
  - Interfaces revealing only method signatures
  - Abstract base classes
- Using collection classes, iterators, and other common library components

Learning Outcomes

1. Design and implement a class. [Usage]
2. Use subclassing to design simple class hierarchies that allow code to be reused for distinct subclasses. [Usage]
3. Correctly reason about control flow in a program using dynamic dispatch. [Usage]
4. Compare and contrast
   - the procedural/functional approach (defining a function for each operation with the function body providing a case for each data variant) and
   - the object-oriented approach (defining a class for each data variant with the class definition providing a method for each operation). [Assessment]
5. Explain the relationship between object-oriented inheritance (code-sharing and overriding) and subtyping (the idea of a subtype being usable in a context that expects the supertype). [Familiarity]
6. Use object-oriented encapsulation mechanisms such as interfaces and private members. [Usage]
7. Use iterators (later in DS topics define).

DS1 Concepts

DS1 Learning Outcomes

Basic Analysis

Concepts

- Differences among best, expected, and worst case behaviors of an algorithm
- Asymptotic analysis of upper and expected complexity bounds
- Complexity classes, such as constant, logarithmic, linear, quadratic, and exponential
- Time and space trade-offs in algorithms
- Big O notation: use

Learning Outcomes

1. Explain what is meant by "best", "expected", and "worst" case behavior of an algorithm. [Familiarity]
2. In the context of specific algorithms, identify the characteristics of data and/or other conditions or assumptions that lead to different behaviors. [Assessment]
3. Determine informally the time and space complexity of simple algorithms. [Usage]
4. List and contrast standard complexity classes. [Familiarity]
5. Give examples that illustrate time-space trade-offs of algorithms. [Familiarity]
6. Use big O notation formally to give asymptotic upper bounds on time and space complexity of algorithms. [Usage]

Fundamental Data Structures and Algorithms

Concepts

- Brute-force algorithms
- Simple numerical algorithms, such as computing the average of a list of numbers, finding the min, max, and mode in a list, approximating the square root of a number, or finding the greatest common divisor
- Sequential and binary search algorithms
- Worst case quadratic sorting algorithms (selection, insertion)
- Worst or average case O(N log N) sorting algorithms (quicksort, heapsort, mergesort)
- Hash tables, including strategies for avoiding and resolving collisions
- Binary search trees: common operations on binary search trees such as select min, max, insert, iterate over tree

Learning Outcomes

1. Describe the implementation of hash tables, including collision avoidance and resolution. [Familiarity]
2. Explain how tree balance affects the efficiency of various binary search tree operations. [Familiarity]

Programming Practices

Concepts

- Basic syntax and semantics of a higher-level language
- Variables and primitive data types (e.g., numbers, characters, Booleans)
- Expressions and assignments
- Simple I/O including file I/O
- Conditional and iterative control structures
- Functions and parameter passing
- The concept of recursion

Learning Outcomes

1. Analyze and explain the behavior of simple programs involving the fundamental programming constructs variables, expressions, assignments, I/O, control constructs, functions, parameter passing, and recursion. [Assessment]
2. Identify and describe uses of primitive data types. [Familiarity]
3. Write programs that use primitive data types. [Usage]
4. Modify and expand short programs that use standard conditional and iterative control structures and functions. [Usage]
5. Design, implement, test, and debug a program that uses each of the following fundamental programming constructs: basic computation, simple I/O, standard conditional and iterative structures, the definition of functions, and parameter passing. [Usage]
6. Write a program that uses file I/O to provide persistence across multiple executions. [Usage]
7. Choose appropriate conditional and iteration constructs for a given programming task. [Assessment]
8. Describe the concept of recursion and give examples of its use. [Familiarity]
9. Identify the base case and the general case of a recursively-defined problem. [Assessment]