CSci 5106: Programming Languages

Fall 2016, Prof. Eric Van Wyk

Instructor: Eric Van Wyk, `evw@umn.edu`, (612) 625-0329, Keller Hall 6-203.
office hours: Monday: 11:15am to 12:15pm, Wednesday: 2:00pm to 3:00pm, or by appointment.

Teaching Assistant: Lucas Kramer, `krame505@umn.edu`.
office hours: TBD. Check the course Moodle site.

Important Dates:

- Class meets on Monday and Wednesday from 9:45am until 11:00am in Keller Hall room 3-111.
- The final exam is Tuesday, December 20 at 1:30pm and will be held in Keller Hall 3-111.
- The dates for two in-class midterm exams are, tentatively, October 19 and November 23.

Course Description: This course provides an introduction to the field of programming languages. You will learn about the various paradigm of programming languages and about the design and implementation issues of high-level programming languages. A goal of this course is for you to develop a mental framework of the design and implementation space for programming languages. By examining different paradigms of programming languages, along with variations in type systems and different languages within the paradigms, you will be able to more quickly and completely understand and evaluate programming languages you know now as well as new programming languages you will encounter in the future.

We will engage a number of questions about programming languages during the semester. These include the following:

- What are the major paradigms (or categories) of programming languages? What are their distinguishing characteristics? How are languages implemented so that we can run programs written in them? What criteria can be used to evaluate programming languages?
- How do we formally specify the syntax and syntactic structure of programs in particular languages? How does a language implementation convert a textual (linear) representation of the program into some abstract representation over which we can perform operations such as type checking, optimization, and code generation?

  Here we study regular expressions and context free grammars as means for specify the lexical and context free syntax of a language.
- What are the characteristics of *imperative* programming languages? How can we formally specify the meaning (semantics) of programs in these kinds of languages?

  Here we will study various notions of semantics: axiomatic semantics, denotational semantics, and attribute grammars.
- How is data typically represented in imperative and functional programming languages?
- How is data passed into and out of functions and procedures? What are the various ways of doing this? How does *scoping* work to bind uses of identifiers to their declarations? How can we implement languages with recursive procedures or functions?

  Here we study call-by-value, call-by-name, call-by-value-result, and copy-in/copy-out parameter passing mechanisms. We also consider static vs. dynamic scoping and how activation records provide a mental model and implementation for procedure invocation.

- What are the key features of object oriented programming languages?

  Here we consider notions of class, object, inheritance, encapsulation, and information hiding. We also study *behavioral subtyping* as a way to properly understand subclass/superclass relationships.

- What are the fundamental characteristics of languages in the functional programming paradigm? How are expressions in these language evaluated? How can types be checked, or better yet, inferred?

  Here we consider higher order functions, parametric polymorphism, and Hindley-Milner style type inference and type checking. We study lazy (non-strict) and eager (strict) evaluation, infinite data structures, and a rewriting semantics for these. We also study various applications of functional programming.

- What is *logic programming*? How is computation in this paradigm structured and specified? How do the *relations* in logic programming compare to the functions in functional programming?

  Here we consider the language Prolog, notions of unification and its use in the underlying search mechanisms of Prolog. We also discuss various applications of logic programming.

We will survey and use many different languages in the imperative, functional, logic and object-oriented paradigms. These include C, Pascal, Java, C++, Standard ML, OCaml, Scheme, Lisp, Haskell and Prolog. Our goal however is not to become proficient in these languages but to learn enough of them to understand their fundamental characteristics.

Prerequisites: To be successful in this course it is important that you are properly prepared to study and learn the material. The formal prerequisite is CSci: 4011 Formal Languages and Automata Theory or "consent of instructor". If, however, you have only satisfied the prerequisites of 4011, namely CSci 1901 and CSci 1902 (or CSci 2041) and have programmed in a few different languages you **may be** prepared to take this course. You are required to see me if you have not satisfied the formal prerequisites for the class.

Materials and Resources:

- Course text: *Programming Languages: Concepts and Constructs* by Ravi Sethi, published by Addison-Wesley, 2nd edition.
- Moodle site: `https://ay16.moodle.umn.edu/course/view.php?id=1837` You should have been automatically provided access to this Moodle page, see the TA if you have not.
- Public repository: `https://github.umn.edu/umn-csci-5106F16/public-class-repo`
- Course notes: Prepared notes on the course material will be provided on the web pages listed above. They are meant to serve as a *guide* to the material and **do not** contain all of the important content. Most material is presented on the whiteboard. **You are expected to take notes during class.**
- We will also read a number of papers and book chapters from other texts. These will be provided electronically as PDF files from the web pages listed above.

  Several of these papers provide the original presentation of some of the founding big ideas in programming languages. Besides providing a technical presentation of the idea, they also serve as a bit of historical perspective on the field of programming languages.

Lecture format: Lectures are designed to be rather interactive and less like a traditional lecture. There will be exercise that we do in class. So come prepared to work. Most lectures will have a large "white board" component that will not be found in the slides. Lectures will present material not in the texts and not on the lecture slides, which will be available on the course web page in various formats.

Engagement: Learning is much more than simply acquiring a collection of facts; it is a process of assimilating knowledge, using it, applying it, and organizing it so that you understand the relationships between different concepts. It is an *active* process, not a *passive* one. Thus it is critical that you are **actively engaged** in the course. What does that mean?

- **Attending class:** My classes involve lots of discussion and in-class exercises. If you are not in lecture, then you miss out on this important way to learn the material. To be actively engaged in the course you will come prepared, having read the assigned reading and doing the exercises to prepare you for class. You will also be an active participant in classroom activities.

  I do not count attendance as part of your grade. It turns out that I do not need to. Those who attend and are engaged *invariably* do much better on assignments and exams that those that do not consistently attend class. The best way to do well in this course is to attend class.
- **Avoiding distraction:** Research has shown that people are very bad at multi-tasking. Your brain cannot do two things well at once and most of out attempts to do two things at once make us slower and, frankly, dumber.

  For this reason, I do not allow cell phones in class. These are to be stored in your bag or in your pocket and not held in your hand or laid on your desk.

  For similar reasons, I do not allow laptops in class. There may be a few occasions in which we want to use them as part of an in-class exercise, but otherwise they are to be put away. The distraction to you is significant but they also distract those around you. (If there is a reason that you feel you need to use your laptop to take notes, then please speak to me about it.)
- You are responsible for routinely checking Moodle for updates and announcements.
- Electronic discussion forums will be available on the Moodle class site. The TA and I will be reading discussions on this and providing our input, but it will primarily be a forum for students to discuss course topics and questions (not solutions) about homework assignments. Asking and answering questions here is an important part of engaging in the class material.

Course work: You will complete several programming and "on paper" assignments to help you engage the questions laid out above and to learn the finer points of the material.

Several of these assignments will be programming assignments. Some of these are to understand the concepts of languages in the different paradigms. Others are to implement solutions to programming language implementation problems, for example, building in interpreter for a simple imperative language based on denotational semantics. Or constructing a scanner and parser to discover the syntactic structure of a textual representation of a program. These will be turned in electronically on `http://github.umn.edu`.

Other assignments are "on paper" assignments. These include using axiomatic semantics to construct proof of partial correctness of programs and hand execution of programs using different methods of parameter passing. Grading illegible homework can be exceedingly time consuming and thus we do not accept work that is not easy to read. It is suggested that you type your answers to the non-programming homework assignments, perhaps using LaTeX, but if you choose not to, your answers must be written clearly and legible. Illegible answers will be marked as incorrect as we cannot grade what we cannot read.

Exams are also an important part of learning and not used solely for evaluation. By properly preparing for an exam you have an opportunity to step back from the specific concepts of individual topics and see how the pieces fit together. The encourage you to do this, your are allowed a "cheat sheet" for the exams on which you can write whatever you like. This sheet must be an $81/2 \times 11$ inch piece of paper. You can use one side of it and it must be hand written. Organizing this document and writing it out by hand is a great way to learn the material. Many students report carefully creating their cheat sheet and then not using it in the exam because they've learned all the material they wrote on it.

We will also spend a considerable amount of time doing exercises in class. These are an important component of the course and thus it is important that you consistently attend lecture. We will collect some of these exercises to ascertain how well students understand the material and also, on occasion, to grade them. But their primary purpose is to help you learn the material in class, often by discussing your solution or questions your fellow classmates.

Course policies
All policies (many presented below) may evolve and change over the course of the semester at the discretion of the instructor. Sometimes issues arise that cannot (or were not) planned for, and I may need some flexibility in handling them.

Required Work and Grading: The exams and homework assignments will draw questions from potentially all of the material in the specified sections of the text and papers, even if this material is not covered in detail, or at all, in the lectures. Also, lectures may also contain information not in the texts or papers, but you will be responsible for this information on the exams and homework as well.

There will be several homework assignments over the course of the semester. These are to be turned in by the **beginning of class** on their due dates or at the specified time.

Numerous in-class exercises will be given and sometime collected after class. Sometimes these are graded but often they are used to determine who well the class, collectively, understands the material. There may also be some short Moodle-based quizzes to be completed before class after you've read the assigned material. These will contribute no more than 10% of the cumulative homework score, which is 5% of your final grade.

Your final cumulative score for the course will be determined based on your scores on homework assignments and exams as follow:

- Cumulative homework score – 40%.
- Mid-term exam 1 - 17.5%.
- Mid-term exam 2 - 17.5%.
- Final exam – 25%. This exam is cumulative.

Different homework assignments will contribute different amounts to your cumulative homework score. This distribution will be determined as the course progresses. You are expected to turn in *all (outside-of-class) homework assignments* if you expect to obtain a passing grade. In class exercises do not count substantially towards your grade and cannot be made up if you miss lecture.

Your final letter grade will be determined by this final cumulative score. In most instances of this course, final grades have been assigned on a scale not unlike the following:

- above 90% - A,
- above 80% - B,
- above 70% - C,
- above 60% - D,
- otherwise - F.

Note that each instance has different exams and homework assignments and thus no fixed scale is always appropriate.

To pass the class your cumulative homework score must represent a passing grade and thus be above 60%. Similarly, to pass the class your cumulative exam score must represent a passing grade and thus be above 60%.

Late assignment and make up exams are generally not accepted or allowed unless previous arrangements have been made with the instructor or are due to a verifiable illness. You should be aware of the University Senate's policy on make-up exams available at
`http://www.policy.umn.edu/Policies/Education/Education/MAKEUPWORK.html`
and their policy on grading available at
`http://policy.umn.edu/Policies/Education/Education/GRADINGTRANSCRIPTS.html`.

Be sure to keep regular track of your accumulating score to make certain that it is correct. You should consult Moodle for this information and make sure that all your scores are correctly recorded. Check with the TA if you find errors. Errors must be reported to no later than 2 weeks after the scores are posted.

Academic Integrity: You are encouraged to discuss homework problems with your fellow students. A large part of solving a problem is getting a precise and complete understanding of what the problem asks. This also helps to resolve any misunderstanding you may have of the problem or unintentional ambiguities in the problem description.

Discussing *answers* to problems, however, is not allowed. The work that you turn in to be graded is to be your own independent work representative of your independent thinking. Your discussions should stop long before you get to details of a solution. If you are still in need of assistance at this point, seek it from the TA or the instructor.

Discussing solutions to problems or copying solutions from others is considered cheating and there are penalties for such action, including:

- On the first offense, no credit will be given on the assignment or exam and a final grade of A or A- will be converted to a B+.
- The second offense results in failing the class.
- All incidents are reported to the Office for Student Conduct and Academic Integrity.

Cheating does not help one learn the material and thus defeats the whole purpose of being in school in the first place. Also, the homework is intended as a warm up for the exams - if you don't learn how to solve the problems by doing the homework then your grades on the exams will surely suffer.

Software is used to detect similar assignments. When flagged, these are investigated by hand.

Incidents of cheating will be reported to the Director of Graduate or Undergraduate Studies in the department and to the appropriate parties at the college and university levels.

The Computer Science and Engineering Department recently formalized it policies on student academic conduct. These policies can be found on the course Moodle web page and you are expected to read, in full, this document. Lack of understanding of these policies does not prevent them from being enforced. Additional departmental information on academic integrity can be found here: `http://www-users.cs.umn.edu/~barry/intro/acad-conduct.html`
You are expected to read and understand both of these documents. The Regent's Policy on Student Conduct, specifically Section IV, Subd. 1. Scholastic Dishonesty, addresses these issues and can be found at `http://regents.umn.edu/sites/default/files/policies/Code_of_Conduct.pdf`

Disability Resource Center: The University is committed to providing all students equal access to learning opportunities. The Disability Recourse Center (DRC) is the campus office that works with students who have disabilities to provide and/or arrange reasonable accommodations.

- Students who have, or think they may have, a disability (e.g. mental health, attentional, learning, vision, hearing, physical or systemic), are invited to contact DRC to arrange a confidential discussion at 612- 626-1333 (V/TTY) or `drc@umn.edu`.
- Students registered with DRC, who have a letter requesting accommodations, are encouraged to discuss accommodations outlined in the letter with the instructor early in the semester.

Mental Health resources: You may experience a range of issues that can cause barriers to learning, such as strained relationships, increased anxiety, alcohol/drug problems, feeling down, difficulty concentrating and/or lack of motivation. These mental health concerns or stressful events may lead to diminished academic performance or reduce your ability to participate in daily activities. University of Minnesota services are available to assist you with addressing these and other concerns you may be experiencing. You can learn more about the broad range of confidential mental health services available on campus via `http://www.mentalhealth.umn.edu`.