# CS 331 – Programming Languages
# Syllabus – Spring 2017

**INSTRUCTOR:** Tom Naps

**OFFICE:** Halsey 214, phone 424-1388

**EMAIL:** naps@uwosh.edu

**OFFICE HOURS:** MWF 9:30 - 10:30, Thur 10:00 - 11:30

**CLASS MEETING TIME:** MWF 1:50 - 2:50

**REFERENCES:**

- An in-progress interactive online book currently being authored by Professor Furcy and myself. This is hosted at `canvas.instructure.com`. Instructions about how you access it have already been emailed to you. Chapters 1 and 3 of the book are complete. The other chapters are presently comprised only of review problems, which are necessary for you to do on a daily basis. At the end of each class, I will tell you which review problems you are responsible for doing. (This information will also be posted on D2L.) Those review problems should then be completed before the start of the next class.
- Handouts, readings, guide-sheets, assignments etc. that are needed for our class sessions will be posted by midnight of the day preceding the class meeting. They comprise a high-level, but not detailed, outline of what we will cover during class. Get a copy of them before class, organize them, take notes on and about them. Handouts that are not liberally saturated with your own explanatory notes will likely prove useless when you need them most.

## Topic Coverage

1. Grammars and formal syntax
2. The functional programming paradigm
3. Recursion in programming
4. Scope and lexical structure of programs
5. The lambda calculus as a formal model of functional languages
6. Writing interpreters for small languages
7. Type systems
8. Evaluation Order and parameter-passing

## Learning Outcomes

- Given an English description of a formal language, the student will be able to construct a context-free grammar that generates this language.
- Given a context-free grammar and the source code for a program, the student will be able to parse the program according to the grammar, and to produce a parse tree of the program or identify syntactic errors in the program.
- Given a context-free grammar in Backus-Naur Form (BNF), the student will be able to convert it to an equivalent, more compact grammar in Extended BNF.
- Given a context-free grammar, the student will be able to determine whether the grammar is ambiguous or not.
- Given a program and a scoping mechanism (static or dynamic), the student will be able to trace the execution and infer the output of the program.
- Given a formal description of an operation, the student will be able to implement it in the functional paradigm using either recursion or a computational pattern such as filtering, mapping, or folding, or a combination thereof.
- Given an imperative program and a set of eager/lazy parameter-passing mechanisms, the student will be able to simulate, for each mechanism, the sequence of updates that take place in memory as the program executes. Given the description of an operation applicable to an infinite data structure, the student will be able to program this operation in a functional language using lazy evaluation.
- Given a functional language with higher-order functions, the student will be able to simulate recursion using the Y combinator.
- Given a working interpreter for a programming language, the student will be able to adapt the interpreter to a similar language with a different concrete syntax.
- Given a working interpreter for an imperative programming language, the student will be able to implement an interpreter for an enhanced language with additional features (such as a new data type or a new language construct), or different semantics (such as a different parameter-passing mechanism).
- Given a problem involving different variants of a data type (e.g., different types of expressions) and different operations on those data (e.g., evaluation, pretty- printing, type-checking, etc.), the student will be able to compare and contrast the procedural/functional approach and the object-oriented approach and to discuss the strengths and weaknesses of each approach. Given a desirable property of a program or a software development process (e.g., programming convenience, ease of prototyping, support for code reuse and code evolution, high run-time performance, etc.), the student will be able to discuss the pros and cons of static checking and dynamic checking with respect to this desirable property. Given a programming language and its type system, the student will be able to define the properties of soundness (or safety) and completeness of this type system, and to compare and contrast weak typing and strong typing.

## Course Grading Policies

Your grade for the course will be based on the following weighted factors:

| **Factor** | **Weight** |
|---|---|
| Homework Assignments (6 - 8) | 45% |
| Exams (three – dates to be announced): | 45% |
| Daily Class Participation | 5% |
| Review problems | 5% |

To get the 45% contribution to your grade from the three exams, I will use the formula:

$$E = 0.10 \times E_{worst} + 0.20 \times E_{best} + 0.15 \times E_{other}$$

where $E_{worst}$ is your worst exam score and $E_{best}$ is your best exam score. At the end of the term, your work in all of these areas will contribute to a numerical grade for the course based on a 100-point scale. Grade cutoff levels on this final scale are:

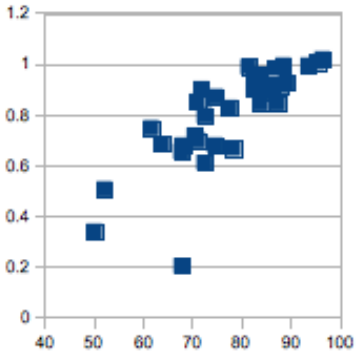
| | | | |
|---|---|---|---|
| A $\geq$ 92 | B $\geq$ 82 | C $\geq$ 72 | D $\geq$ 62 |
| A- $\geq$ 90 | B- $\geq$ 80 | C- $\geq$ 70 | D- $\geq$ 60 |
| B+ $\geq$ 88 | C+ $\geq$ 78 | D+ $\geq$ 68 | F $<$ 60 |

## FAQ

**Do I have to come to class?** You are expected to arrive prepared to ALL the course sessions. Furthermore you are expected to participate in the classroom discussions and activities to the best of your abilities. This includes your taking advantage of the opportunity to present your solution to a review problem when called upon to do so. Your doing this will not only ensure that you get the full 5% "class participation" component factored into your grade; more importantly the learning that transpires during these activities will help tremendously in preparing you for what is on the exams. It is difficult to envision a student missing and/or arriving unprepared to a number of the class sessions and still succeeding in the course.

**What if I'm late in completing and submitting a homework assignment?** It will be accepted but will be penalized at the rate of 10% of point value the first day late, *an additional* 20% the second, *an additional* 30% the third . . .

**What if I'm late in completing the daily review exercises?** You will get zero credit for that set of review exercises. Although the review problems only count 5% of your grade, the following correlation from a previous course between review-problem-percentage (on a 0 to 1.0 vertical scale) and overall percent in the course (on a horizontal scale of 0 to 100) is indicative of their true importance.




We will always discuss the review problems assigned on a given day at the beginning of the next class. If you have participated in class the day the review problem was distributed, have made a good faith effort to work on the review problem, and are "stuck" on it, I will be more than happy to help you with it if you come my office anytime within three days after the review problem has been assigned. After those three days, *because you have made the choice to not learn efficiently*, you are on your own in terms of grappling with these review problems. In particular, I will not entertain questions about past review problems as an exam approaches.

**If I miss an exam, can I make it up?** If you are unable to take a scheduled exam, it may be possible to take a make-up exam provided that you do BOTH of the following, which are then subject to my approval:

- Make arrangements prior to the scheduled exam (for last minute emergencies, telephone me at 424-1388 or leave a message at the computer science office, 424-2068). No after-the-fact notifications will be accepted . . . *AND*

- Have a written medical excuse signed by the attending physician OR have a note of justification from the Dean of Students Office.

Only one make-up exam will be given. It will be a rigorous comprehensive exam given at an arranged time during the last week of the semester.

**Can I work with others on the homework assignments?** You have the choice to work on your own or with *one* partner, in effect working in *pair learning/programming* style. (See research by A. Cockburn and L. Williams indicating that, when done conscientiously, working in paired environments will slightly increase the time you put in and significantly improve the quality of the work.) You are not allowed to collaborate with anyone else for assignments. If you decide to work with a partner, here is the way that it must be done:

- *Both* you and your partner must "declare" via emails sent to me *three days* before an assignment is due that you will be working together. Once that declaration is made, it will remain in effect until you email me again that you no longer wish to work as partners.
- If you decide to work with a partner, only one of you will submit the assignment, and that is what I will grade. You will both receive the same grade.
- You may not share or even discuss your work with anybody but your partner unless you can live with a zero and the other potential academic sanctions of cheating (see the UWO Student Discipline Code, Chapter UWS 14).

If you decide to work on your own, then the third bullet point above also applies (minus the partner).