

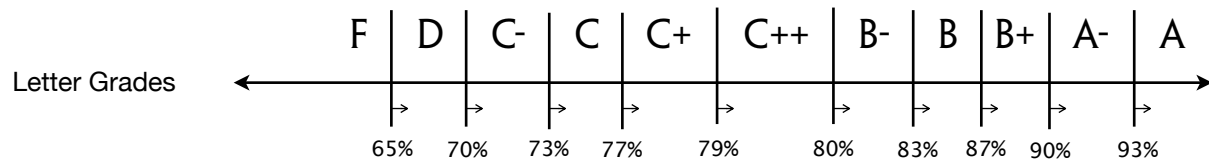
# Language Study: Erlang

CMSC 233 • Fall 2010

## -Background

When and Where	LT-001 Tuesday afternoons 3:30pm through 4:45pm LT-135 Thursday afternoons 3:30pm through 4:45pm	
Required Text	<i>Programming Erlang</i> by Joe Armstrong - ISBN: 978-1-93435-600-5	
Web Site	Labouseur.com/courses/erlang	
Instructor	Alan G. Labouseur LT 101 (office hours posted)	Alan.Labouseur@Marist.edu alan@labouseur.com 845-575-3000 x2831 Marist phone 845-440-1102 home office phone

## -Grading



You can earn up to 1000 points over the course of the semester, broken down over the following areas:	Homework	15%	150 points: 3 at 50 points each
	Mid-term Exam	15%	150 points
	Mid-term Project	20%	200 points
	Final Exam	15%	150 points
	Final Project	20%	200 points
	Attendance	5%	50 points for <b>consistently</b> showing up
	Participation	5%	50 points for <b>constructive</b> participation
	Laziness Adjustment	3%	30 points for <b>not</b> being lazy
	Whining Adjustment	2%	20 points for <b>not</b> whining

## -Objectives and Assessment

Assessment methods include assignments, quizzes, exams, discussions, presentations, and projects.	<ol style="list-style-type: none"><li>1. Develop the skills to design functional and concurrent programs.</li><li>2. Be able to construct parallelizable functions that work together.</li><li>3. Demonstrate the core concepts of functional programming.</li><li>4. Demonstrate the core concepts of distributed programming.</li><li>5. Provide students an opportunity to develop software systems over the course of the semester, where they have to live with their past mistakes and shortcuts, or fix them. Either will teach a valuable lesson.</li><li>6. Development is only half the battle. Debugging is a critical skill, and is stressed in this course.</li><li>7. Students will get practice in finding some answers for themselves. Capable problem solvers never stop learning.</li></ol>
---	---

# Language Study: Erlang

CMSC 233 • Fall 2010

## -Proposed Schedule

#	Week	Ch	Topic	Required
1	31-Aug	2, 7	<i>Tuesday Theory:</i> Introduction, Expectations, and Goals / our Book <i>Thursday Practice:</i> Installing Erlang / The shell / Interactive math / Vars	<i>Attention</i>
2	7-Sep	2	<i>Tuesday Theory:</i> Pattern matching / Tuples / Lists / Strings <i>Thursday Practice:</i> Lab time with pattern matching, tuples, lists, strings	<i>Practice</i>
3	14-Sep	3	<i>Tuesday Theory:</i> Sequential Programming / Modules / Functions <i>Thursday Practice:</i> Lab time with Modules and Functions	<b>Hw 1</b>
4	21-Sep	3	<i>Tuesday:</i> Guest Speakers from Morgan Stanley <i>Thursday Practice:</i> Guards / Lab time with guards and recursion	<i>Practice</i>
5	28-Sep	3, 4, 5	<i>Tuesday Theory:</i> Case & If / Anonymous Functions <i>Thursday Practice:</i> Lists and higher-order functions / BIFs / Lab time	<b>Hw 2</b>
6	5-Oct	1 - 5	<i>Tuesday Theory:</i> Mid-term Exam essays <i>Thursday Practice:</i> Mid-term Exam practical	<i>Expertise</i>
7	12-Oct	-	<i>Tuesday:</i> Active Learning Activities for Mid-term project <i>Thursday:</i> Mid-term Project due	<b>Mid-Term Project</b>
8	19-Oct	1 - 5	<i>Tuesday Theory:</i> Review mid-term... um... Issues / Reevaluate plan <i>Thursday Practice:</i> Review recursion and anonymous functions	<i>Practice</i>
9	26-Oct	13	<i>Tuesday Theory:</i> Reading from files / Funs that return Funs <i>Thursday Practice:</i> Guest Speaker - "Professional Software Development"	<i>Practice</i>
10	2-Nov	7 - 8	<i>Tuesday Theory:</i> Concurrency and Concurrent Programming <i>Thursday Practice:</i> Lab time with spawn, send, and receive	<b>Challenge</b>
11	9-Nov	8	<i>Tuesday Theory:</i> More Current Programming <i>Thursday Practice:</i> Lab time with spawn, send, and receive / Work on Hw3	<i>Practice</i>
12	16-Nov	9	<i>Tuesday Theory:</i> Distributed Systems in Erlang <i>Thursday Practice:</i> Lab time with distributed programs	<b>Hw 3</b>
13	23-Nov	-	<i>No class meetings - Thanksgiving</i>	<i>Practice</i>
14	30-Nov	9 - 10	<i>Tuesday Theory:</i> Distributed Programming / Remote spawning <i>Thursday Practice:</i> Lab time with distributed programs	<i>Practice</i>
15	7-Dec	-	<i>Tuesday:</i> Final Exam part one <i>Thursday:</i> Final Exam part two	<i>Expertise</i>
16	14-Dec	-	<b>Tuesday at 3:30pm - Final Project Presentations</b>	<b>Final Project</b>

# Language Study: Erlang

CMSC 233 • Fall 2010

## -Policies

---

Tests	Tests cover material presented up to the class in which the test is given. No makeup tests will be given. Ever. If you anticipate missing a test, make arrangements with me in advance to hand in the exam prior to its due date.
Homework	All assignments must be handed in and/or uploaded at the beginning of class on the day they are due. If you're going to miss a class (which is, itself, a bad idea) arrange to submit your homework on schedule anyway.
Late Submissions	No assignments will be accepted late. Ever. The reason is that we may discuss some possible solutions in the class in which it's due. Discussion is an important part of the learning process, and once we cover the assignment in class, you clearly cannot hand it in after that.
Attendance and Communication	Students are expected to attend every class. Attendance may or may not be officially recorded, but it will always be noted, and I never forget. The official means of communication for this course will be in-class announcements. Missing class is no excuse for failure to act as required by these announcements.
Etiquette	Students are expected to be on-time for every class, return to class after breaks, and keep their cell phones turned off during class.
Appealing Grades	<p>I have an appeals process to handle any questions you might have about fairness related to my grading of your work. I will address each and every one of your concerns. To that end, and in order to be fair and efficient, you must to write a letter of appeal if you want me to alter your grade.</p> <p><i>Rules for Submitting an Appeal</i></p> <ul style="list-style-type: none"><li>• Appeals must be in the form of a neatly written letter.</li><li>• Appeals must be on a separate paper and stapled to the work in question.</li><li>• Every appeal (if there is more than one) requires its own paragraph.</li><li>• Appeals are due the next class period after the work is returned to you.</li><li>• Appeals must be very specific.</li><li>• Appeals must be content-based, not personal or emotional.</li><li>• Insufficient time is not a basis for an appeal.</li><li>• You must communicate what action you would like me to take, for instance give full credit, add points, etc.</li></ul> <p>This process empowers students, advances learning, and moves students toward academic maturity. As such, it benefits both the teacher and the student. Further, students are given a method to argue their points in an appropriate manner and explain their reasoning, while the teacher has an opportunity to learn whether or not he has understood students' reasoning.</p>

# Language Study: Erlang

CMSC 233 • Fall 2010

## Policies

---

### Taking Notes

You are expected to take notes in class. Note what we talk about in class: what I say, what you say, what others say. Everything covered in class or assigned as homework is fair game for tests and quizzes. I do not often (if ever) distribute notes, so you must take them on your own; it's part of the learning process. To that end, a good way to prepare for an exam is to rewrite your notes, thereby reinforcing and organizing the material.

### Lab Work

It's entirely possible that you may encounter some topics in lab prior to our discussing them in class. **Don't panic!** Our lab instructors are fine teachers as well as experts in this field. As such, they are extremely well equipped to introduce you to new topics. Then, when we get there in class, you can amaze me with your knowledge.

### Learning to Learn

Capable professionals know how to solve problems, even -- perhaps most especially -- in the absence of complete knowledge. This is a large part of what I want to teach you. To that end, I will encourage and at times require you to practice finding things out for yourself. There will be occasions when you need to look things up and find things out *on your own* to complete an assignment. This is an important skill, and one that will serve you for the rest of your life, so we might as well begin practicing it now.

### Students with Disabilities

Any student requesting or wondering about accommodations based on a disability should see the fine folks at the Office of Special Services in Donnelley 226 and online at [www.marist.edu/specialservices](http://www.marist.edu/specialservices).

### Safety and Security

***If you see something, say something.***

Report all emergencies or suspicious activity or persons to the Office of Safety and Security

Emergency - x5555 or 845-575-5555

All Other Calls - x2282

Outside Line 845-471-1822

SNAP Escort Service - x 7627 (SNAP)

All classrooms have a phone capable of calling Security in an emergency. The Building name and room number is posted on the inside of all classrooms. Tell Security your location and nature of the problem.

Classrooms have door locks on the inside to prevent entry of intruders. (Do not use these to keep your professors out. They hate that.)

Emergency Information placards have been placed in all classrooms close to the phone. Read them and note the evacuation routes.

Close all doors as you leave.

# Language Study: Erlang

CMSC 233 • Fall 2010

## Policies

---

### Guidelines for Grading Programs and Projects

All programs and applications must be free of syntax errors to receive any credit. Programs that ((compile or interpret) and execute) cleanly but contain logic errors will be graded based on the severity of the errors and how well your work demonstrates your approach to solving the problem.

When evaluating your programming assignments I will ask myself the following questions about your program:

- Is it correct? I.e., free from faults in specification, design, and implementation?
- Is it usable? About the interface...
  - ▶ Is it “clean” and well-organized? Would Mr. Monk be proud?
  - ▶ Does it obey the Laws of Least Astonishment?
  - ▶ Is it accurate?
  - ▶ Is it easy to use?
- Is it reliable and robust? I.e., can it perform its functions without breaking down, even given unexpected input or other circumstances?
- About the readability and maintainability of the source code...
  - ▶ Are the comments plentiful, clear, meaningful, and helpful?
  - ▶ Are the identifier names accurate, clear, and meaningful?
  - ▶ How is its functional architecture?
  - ▶ Does it compile or interpret cleanly?
- About the Design...
  - ▶ Is the solution well-designed?
  - ▶ Is it re-usable?
  - ▶ Does it illustrate the points made and principles discussed in class?
  - ▶ Have any lazy shortcuts been taken?
  - ▶ Can the program be unit and system tested?
- About the results, are they accurate? I.e., are the qualitative outputs free of error?
  - ▶ Does it perform as assigned?
  - ▶ Is the output well-formatted and meaningful?

Remember, neatness and style count. If you hand in a program that works, but that does not adhere to reasonable style standards, is inadequately commented, or is poorly designed, you will be penalized. Good habits are important and I want you to develop some.

---

### Fire Alarms

Everybody must immediately evacuate the building when the fire alarm sounds. Do not use elevators during a fire alarm. Once outside, get to a safe distance from the building and do not re-enter until given the “all-clear” from the Fire Department or Security Officer. (Two fire drills are conducted each semester, so you have that to look forward to.)

# Language Study: Erlang

CMSC 233 • Fall 2010

## -Policies

---

### Academic Honesty

As a part this class, I will uphold and **vigorously enforce** the general policies of this institution on academic honesty and plagiarism. All examinations, papers, projects, and homework assignments are subject to the usual standards of academic honesty as described in the Student Handbook and/or other related publications.

All work must be your own. Period. End of story. This applies to homework, tests, quizzes, projects... anything and everything you do for this class. You are free to use reference material (including but not limited to text books and example code or other resources you find in person or online) as a guide or for inspiration. But you must cite all your sources and make the proper references in your work. Further, you may not under any circumstances copy even the smallest part of those materials and present it as your own work. Any violation of this policy will result in immediate dismissal from the class with a failing grade. There will be no second chances.

Furthermore, I expect my students to behave in a manner appropriate to Computer Science and Information Technology professionals. Professional ethics **demand** that you embrace traditional “thou shall not cheat” behavior, and also that you soundly reject additional forms of dishonesty and abuse which are uniquely possible working with computers.

Remember: Allowing someone to copy your work is every bit as dishonest as copying someone else's, and will be treated just as harshly.

Any violation -- actual or perceived (in my sole discretion) -- of this Academic Honesty policy will result in one or more of the following actions in addition to any other forms of recourse available as specified by the Student Handbook:

- You will be ejected from the course with a failing grade.
- A letter will be sent to your department chair, your Dean, and the president of the college.
- And more. (And worse!)

The bottom line is that I expect you to conduct yourself as a person of integrity. This means that **plagiarism in any form is completely unacceptable**. You are soon-to-be a computing professional, and I encourage you to consult the ACM professional code of ethics. See [www.acm.org/about/code-of-ethics](http://www.acm.org/about/code-of-ethics).

See also <http://www.labouseur.com/courses/honesty>