**ABET Course Syllabus – CS3035**

|  |  |  |  |
| --- | --- | --- | --- |
| **Code** | CS3035 | **Credits** | 3 |
| **Title** | Programming Language Paradigms | **Coordinator** | Russ Abbott & Mark Sargent |

**Course Information**

1. **Catalog Description:** Capabilities and styles of various programming languages; functional programming; concurrent/reactive programming; constraint (logic) programming; rule-based programming; aspect-oriented programming; domain-specific languages. Graded ABC/NC.
2. **Prerequisites:** CS2013, CS2148;
3. **Contact Hours:** Lecture 2 hours, Laboratory 3 hours /week
4. **Required/Elective:** This course is a required course in the BS program.

**Textbook**

 No formal textbook. Textbooks and web resources assigned by the instructor.

**Course Goals**

The Student Learning Outcomes that are addressed by the course are:

*SLO1. Students will be able to apply concepts and techniques from computing and mathematics to both theoretical and practical problems.*

*SLO2. Students will be able to demonstrate fluency in at least one programming language and acquaintance with at least three more.*

*SLO3. Students will have a strong foundation in the design, analysis, and application of many types of algorithms.*

*SLO5. Students will have the training to analyze problems and identify and define the computing requirements appropriate to their solutions.*

*SLO8. Students will have the knowledge, skills, and attitudes for lifelong self-development.*

Other outcomes of instruction. At the end of the course, students are able to

* Write programs in a variety of programming languages.
* Have the skills to learn new programming languages.
* Write programs that use features from a range of programming strategies and technologies such as functional programming, constraint programming, rule-based programming.
* Explain how these strategies and technologies work and how using them differs from implementing the same or similar functionality directly in imperative programming languages.
* Learn new and unfamiliar programming strategies and technologies.
* Develop well-honed and sophisticated critical thinking skills.
* Use inductive and deductive reasoning to reach well-supported conclusions and develop cogent arguments.

**Topics covered**

Topics to be selected from the following.

* General programming language principles (language agnostic):
	+ Programming domains, evaluation criteria for languages, computer architecture influence on language design, language categories/paradigms, implementation methods
	+ Variables and the concepts of binding, scope, referencing environments, named constants
	+ Data types: primitive types, character/string types, enumeration types, array types, associative arrays (dictionaries)
	+ Fundamentals of subprograms, design issues of subprograms, local referencing environments, parameter-passing methods, subprograms as parameters, indirect calls to subprograms, design issues for functions, overloaded and generic subprograms, closures, call stacks, stack-dynamic local variables, activation records and record instances
* Functional programming
	+ Basics, immutability, lists and list comprehensions
	+ Pattern matching
	+ Recursion
	+ Curried functions, higher-order functions, maps, filters, lambdas, folds, function composition, monads
* Logic programming
	+ Predicate logic
	+ Knowledge bases and queries
	+ Constraint programming
* Programming Patterns: Strategy, Observer, Factory, and other programming patterns