Arab Academy for Science and Technology and Maritime Transport
Computer Science Curriculum
Course Syllabus

Course Code: CS449	**Course Title:** Functional Programming	**Classification:** E	**Coordinator's Name:** Prof. Dr. Aliaa Youssif	**Credit Hours:** 3
Pre-requisites: CS445 (Structure of Programming Languages)	**Co-requisites:** None	**Schedule:** Lecture: 2 hours Tutorial-Lab: 2 hours		
Office Hours:				

Course Description:

Course Description:

The purpose of this course is to introduce the theory and practice of *functional programming (FP)*. The characteristic feature of FP is the emphasis on *computation as evaluation*. The traditional distinction between program and data characteristic of *imperative programming (IP)* is replaced by an emphasis on classifying expressions by *types* that specify their applicative behavior. Types include familiar (fixed and arbitrary precision) numeric types, tuples and records (structs), classified values (objects), inductive types such as trees, functions with specified inputs and outputs, and commands such as input and output. Well-typed expressions are evaluated to produce values, in a manner that is guaranteed to be type-safe. Because functional programs do not cause side-effects we can take advantage of simple mathematical principles in reasoning about applicative behavior and analyzing the runtime properties of programs.

Textbook:

Thompson S., *Haskell: The Craft of Functional Programming*, Addison-Wesley.

References:

- Pratt and Zelkowitz, *Programming Languages: Design and Implementation*, 4th Edition, Prentice Hall.
- R. Sebesta, *Concepts of Programming Language*, 10th Edition, Addison-Wesley.
- Ullman*, Elements of ML Programming, ML97 Edition,* 2nd Edition, Prentice Hall.
-

Course Objective/Course Learning Outcome:	Contribution to Program Student Outcomes:
1 Understand basic functional programming techniques.	(SO1) Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions.
2 Design programs using types.	(SO2) Design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline.
3 Develop programs using mathematical techniques for verification and analysis.	(SO6) Apply computer science theory and software development fundamentals to produce computing-based solutions
4 Use of abstract types and modules to structure code.	

5 Exploit parallelism in applications.	
Course Outline: 1. Introducing functional programming 2. Basic types and definitions 3. Data types, tuples and lists 4. Programming with lists 5. I/O in Haskell 6. Reasoning about programs 7. Generalization: patterns of computation	8. Higher-order functions 9. Overloading, type classes and type checking 10. Algebraic types 11. Abstract data types 12. Lazy programming 13. Programming with monads 14. Domain-Specific Languages
Grade Distribution: 7th Week Assessment (30%) 12th Week Assessment (20%) Year Work (10%) Final Exam (40%)	

Policies:

Attendance:
AASTMT Education and Study Regulations (available at aast.edu)

Academic Honesty:
AASTMT Education and Study Regulations (available at aast.edu)

Late Submission:
Late submissions are graded out of 75% (1 week late), 50% (2 weeks late), 25% (3 weeks late), 0% (more than 3 weeks late)