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

Course Code: CS445	**Course Title:** Structure of Programming Languages	**Classification:** **R**	**Coordinator's Name:** **Lecturer:**	**Credit Hours:** 3
Pre-requisites: CS311 (Theory of Computation) CS321 (Systems Programming)	**Co-requisites:** None	**Schedule:** Lecture: 2 hours Tutorial-Lab: 2 hours		
Office Hours:				

Course Description:
A concise introduction to the essentials of imperative programming languages, focusing on principles rather than specifics, while giving examples from many programming languages. The course also covers fundamental issues in language design. Overview of programming paradigms and type systems. Models of execution control: Order of evaluation of sub-expressions; conditional execution; iteration; exceptions and exception handling. Basic concepts of functional programming are also emphasized.

Textbook:
R. Sebesta, *Concepts of Programming Languages*, Addison- Wesley.

References:

Apply computer science theory and software development fundamentals to produce computing-based solutions.

H. Bal, D. Grune, *Programming Language Essentials*, Addison-Wesley.

Course Objectives:	**Contribution to Program Student Outcomes:**
1. Be familiar with several language paradigms	SO2: An ability to use current techniques, skills, and tools necessary for computing practice.
2. Understand how languages relate to different application domains.	SO2: An ability to use current techniques, skills, and tools necessary for computing practice. SO6: Apply computer science theory and software development fundamentals to produce computing-based solutions.
3. Understand the design space of programming languages, including concepts and constructs from past languages as well as those that may be used in the future.	SO2: An ability to use current techniques, skills, and tools necessary for computing practice. SO6: Apply computer science theory and software development fundamentals to produce computing-based solutions.
4. Be able to develop a critical understanding of the programming language we use by being able to identify and compare the same concept as it appears in different languages.	SO2: An ability to use current techniques, skills, and tools necessary for computing practice. SO6: Apply computer science theory and software development fundamentals to produce computing-based solutions.

Course Outline:

1. Introduction and Preliminaries
2. Describing Syntax and Semantics
3. Lexical and Syntax Analysis Part I
4. Lexical and Syntax Analysis Part II
5. Names, Bindings, Type Checking, and Scopes
6. **Data Types**
7. **7th Week Exam**

8. Expressions and Assignment Statements
9. Expressions and Assignment Statements (cont.)
10. Statement-Level Control Structures
11. Subprograms
12. **12th Week Exam**
13. Subprograms (cont.)
14. Projects
15. Revision
16. **Final Exam**

Grade Distribution:

7th Week Assessment (30%):
Exam (20%) + Homework Assignments and/or quizzes 10%

12th Week Assessment (20%):
Exam (15%) + Assignments 5%

Year Work (10%):
Course Project (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)