**SUBJECT DESCRIPTION FORM**

---

**Subject Title:** Programming Languages, Theory and Applications **Subject Code:** COMP 328

**Number of Credits:** 3 **Hours Assigned:** Lecture 35 hours
Tutorial/Lab 14 hours

---

**Pre-requisite:** COMP 201, COMP 305 **Co-requisite:** Nil **Exclusion:** Nil

---

**Objectives:**

- To provide students with an understanding of the structure and design principles of programming languages;
- To develop skills in describing, analyzing, and using the features of programming languages.

**Student Learning Outcomes:**

After taking this subject, the students should be able to:
*Professional/academic knowledge and skills*
(1) understand the general language design principles;
(2) understand typing, variable binding, parameter passing and operation of recursive procedure calls;
(3) aware of the program compilation process, the concepts behind and the tools to use;
(4) evaluate the design of a given programming language for the application at hand;

*Attributes for all-roundedness*
(5) solve problems independently;
(6) think critically for a specific design and the rationale behind.

**Alignment of Programme Outcomes:**
Programme Outcome 1: This subject contributes to having students practice their writing skills with project document and report writing.
Programme Outcome 4: This subject contributes to developing student critical thinking through tutorial and lab exercises on solving problems. They will also practice more in written assignments, programming exercises, and project.
Programme Outcome 5: This subject contributes to problem solving with programming skills through lab exercise and project with proper design and implementation.
Programme Outcome 7: This subject contributes to team work with group-based project for students to practice team spirit.

---

**Syllabus:**

| Topic | Duration of Lectures |
|---|---|
| **1. Programming language paradigms**<br>Overview of existing programming languages, e.g. Fortran/Pascal/C/C++, PERL/Python/Ruby, Java, Visual Basic, Lisp, E-mycin, Prolog, Linda; programming paradigms: conventional, script, object-oriented, event-driven, | 5 |

| | |
|---|---|
| functional, rule-based, logic/declarative, concurrent/tuple-space. | |
| **2. Principles of language design**<br>Design principles, e.g. zero-one-infinity principle, information hiding principle; language syntax; language semantics; typing and binding; side effects. | 2.5 |
| **3. Syntax**<br>Abstract and concrete syntax; Backus-Naur form; regular and context free grammars; parse trees. | 5 |
| **4. Names and types**<br>Role of names and types; types operations; functions, recursive types, reference and array types; strong versus weak-typing systems (e.g. Pascal/Java versus C); type checking: static, dynamic; type equivalence: by name, structural; type overloading, coercion and polymorphism; type inference. | 7.5 |
| **5. Block-structured languages**<br>Concept of blocks; environment; scope and visibility of variables; static and dynamic scoping; run-time stack; procedure call; parameter passing semantics; activation records and recursion. | 5 |
| **6. Programming language translation**<br>Concepts of compilers and interpreters; lexical analysis; syntax analysis; symbol table; use of Lex and Yacc; issues of code generation. | 10 |
| **Total** | **35** |

**Laboratory Experiment:** Explore simple programming in different types of languages

**Case Study:** Features and applications of a less-than-common programming language

---

**Method of Assessment for Learning Outcomes:**

| Assessment method / task | % weighting | Intended subject learning outcomes to be assessed (Please check as appropriate) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | 1 | 2 | 3 | 4 | 5 | 6 | | | |
| Assignments | 55 | x | x | x | | | x | | | |
| Lab exercises | | | x | x | | | | | | |
| Project | | | x | x | | | | | | |
| Mid-term | | x | x | x | x | x | x | | | |
| Examination | 45 | x | x | x | x | x | x | | | |
| Total | 100 | | | | | | | | | |

---

**Textbooks:**

1. Robert W. Sebesta, Concepts of Programming Languages, Seventh Edition, Addison

Wesley, 2006.
2. Allen B. Tucker and Robert E. Noonan, Programming Languages: Principles and Paradigms, Second Edition, McGraw-Hill, 2007.
3. Alfred V. Aho, Ravi Sethi and Jeffrey D. Ullman, Compilers: Principles, Techniques, and Tools, Second Edition, Pearson/Addison-Wesley, 2007.

**Reference Books:**

4. Franklyn A. Turbak and Mark A. Sheldon, Design Concepts in Programming Languages, MIT Press, 2008.
5. Dick Grune, Modern Compiler Design, John Wiley, 2000.
6. John C. Mitchell, Concepts in Programming Languages, Cambridge University Press, 2003.
7. Bruce J. MacLennan, Principles of Programming Languages: Design, Evaluation and Implementation, Third Edition, Oxford University Press, 1999.
8. Ravi Sethi, Progamming Languages: Concepts and Constructs, Second Edition, Addison Wesley, 1996.