<table>
<thead>
<tr>
<th>Dept Number</th>
<th>CS 311</th>
<th>Course Title</th>
<th>Theory and Implementation of Programming Languages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semester Hours</td>
<td>3</td>
<td>Course Coordinator</td>
<td>Adil G Ibrahim</td>
</tr>
<tr>
<td>Catalog Description</td>
<td>Introduction to the theory and implementation of programming languages including finite automata, regular grammars, lexical analysis, parsing, syntax-directed translation, semantic analysis, binding variables, data types, static and dynamic scope, subprograms, abstraction, and concurrency. Study of object-oriented, functional, and logic programming languages. Lab work is essential.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Textbooks**


**References**


**Course Learning Outcomes**

- To obtain background on compilers and language compilation
- To understand the basics of the theory of computing applied to develop programming languages
- To learn the features and capabilities those are available in programming languages
- To understand the issues in implementing various programming language features
- To learn the effect of languages on problem solving and programming process

**Assessment of the Contribution to Student Outcomes**

<table>
<thead>
<tr>
<th>Outcome</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assessed</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Prerequisites by Topic**

CS 220 with a grade of C or better
### Major Topics Covered in the Course

1. Introduction: domains, language evaluation criteria, language categories, implementation methods  
   {3 classes}
2. Syntax and semantics: formal methods of describing syntax, attribute grammars, dynamic  
   semantics {6 classes}
3. Finite automata: deterministic and nondeterministic finite automata, regular grammars  
   {5 classes}
4. Lexical and syntax analysis: recursive-descent parsing, bottom-up parsing {5 classes}
5. Variables: names, binding, types, scope, lifetime {2 classes}
6. Basic data types: implementations of integers, strings, etc. {2 classes}
7. Expressions: operators, assignment, precedence, associativity, side effects, overloading, coercion  
   {2 classes}
8. Subprograms: procedural abstraction, generic functions, parameter passing, recursion  
   {2 classes}
9. Abstract data types: data abstraction, user-defined data types, encapsulation, information hiding  
   {2 classes}
10. Concurrency: monitors, threads {2 classes}
11. Exception and event handling {2 classes}
12. Object-oriented programming: basic features, alternative models, implementation requirements  
    {3 classes}
13. Functional and logic programming: clips, lisp, scheme {4 classes}