Dept Number | CS 220 | Course Title | Programming with Data Structures
--- | --- | --- | ---
Semester Hours | 3 | Course Coordinator | Norman Carver
Catalog Description | Advanced programming, data structures and algorithm design. Topics include advanced language features, data abstraction and object-oriented programming, recursion, stacks, queues, linked lists, trees and graphs, sorting and searching. The course meets for three lecture hours and two laboratory hours per week.

Textbooks


References

Course Learning Outcomes

- Understand structured programming techniques in-depth.
- To learn data abstraction and object-oriented programming.
- To learn the fundamental data structures including stacks, queues, linked lists, and trees.
- To learn sorting and searching techniques and their analysis.
- To obtain a good foundation for further study in computer science.

Assessment of the Contribution to Program 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></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Prerequisites by Topic

CS 202 and CS 215 each with a grade of C or better.
Major Topics Covered in the Course

1. Review of programming; arrays, pointers, structures {3 classes}
2. Programming methodology
   Design techniques: in-depth treatment of procedural and data abstraction, further emphasis on
top-down design, choice of data structures
   Coding: additional emphasis on programming style, structured programming, and
documentation, information hiding
   Correctness: testing and test data, testing end cases, debugging techniques, verification of
   algorithms, invariants {3 classes}
3. Data abstraction and object-oriented programming: levels of abstraction; polymorphism,
inheritance, encapsulation {2 classes}
4. Pointers and dynamic allocation: pointer variables; dynamic allocation; pointers as parameters
   {5 classes}
5. Implementation of data structures: lists and linear structures; stacks and queues; trees and
   graphs {14 classes}
6. Recursion
   Implementation: memory and time considerations; simulating recursion
   Efficiency considerations: recursive vs. iterative solutions {14 classes}
   Searching: linear search – review of linear search, searching linked lists, analysis
   Binary search: review of binary search of arrays, binary search trees, analysis {6 classes}
7. Searching and sorting: linear search; binary search; introduction to formal analysis of
   algorithms
   \( N^2 \) sorts: analysis of bubble sort, insertion sort, and selection sort
   \( N\log N \) sorts: quick sort, merge sort, analysis of these sorts {7 classes}