**Nirma University**

**Institute of Technology**

**Computer Science and Engineering Department**

**Course Policy**

**B.Tech. (Computer Science and Engineering)**

**Semester: V , Academic Year: 2020-21, Term: Odd**

|  |  |  |
| --- | --- | --- |
| **Course Code & Name** | **:** | 2CS503 Design and Analysis of Algorithms |
| **Credit Details** | **:** | [2 1 2 4] |
| **Course Co-ordinator** | **:** | Prof. Dhaval S. Jha |
| **Contact No. & Email** | **:** | Ext: - , dhaval.jha@nirmauni.ac.in |
| **Office** | **:** | D-306 |
| **Visiting Hours**  **Course Blog** | **:**  **:** | 11 am to 4 pm  [**https://2ce339avm.wordpress.com/**](https://2ce339avm.wordpress.com/) |
| **Course Faculty** | **:** | Dr. Sanjay Garg |
| **Contact No. & Email** | **:** | sgarg@nirmauni.ac.in |
| **Office** | **:** | PG-3rd floor |
| **Visiting Hours**  **Course Blog** | **:** | 11 am to 4 pm  [**https://2ce339avm.wordpress.com/**](http://2ce339avm.wordpress.com/) |

**The Program Educational Objectives of B. Tech. Program are:**

**PEO I:** To prepare graduates who will be successful professionals in industry, government, academia, research, entrepreneurial pursuit and consulting firms.

**PEO II:** To prepare graduates who will contribute to society as broadly educated, expressive, ethical and responsible citizens with proven expertise.

**PEO III:** To prepare graduates who will achieve peer-recognition; as an individual or in a team; through demonstration of good analytical, design and implementation skills.

**PEO IV:** To prepare graduates who will thrive to pursue life-long learning to fulfill their goals.

**Program Outcomes of B. Tech. Program are:**

**PO1 :** an ability to apply knowledge of mathematics, science and engineering in practice

**PO2 :** an ability to identify, critically analyze, formulate and solve engineering problems with comprehensive knowledge in the area of specialization

**PO3 :** an ability to select modern engineering tools and techniques and use them with appropriate skills

**PO4 :** an ability to design a system and process to meet desired needs within realistic constraints such as health, safety, security and manufacturability

**PO5 :** an ability to contribute by research and innovation to solve engineering problems

**PO6 :** an ability to understand the impact of engineering solutions in a contemporary, global, economical, environmental, and societal context for sustainable development

**PO7 :** an ability to function professionally with ethical responsibility as an individual as well as in multidisciplinary teams with positive attitude

**PO8 :** an ability to communicate effectively

**PO9 :** an ability to appreciate the importance of goal setting and to recognize the need for life-long reflective learning

**Mapping of Course with POs:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Course Name** | **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6** | **PO7** | **PO8** | **PO9** |
| **DAA** | √ | √ | √ | √ | √ | √ |  |  |  |

**Introduction to Course:**

The first step towards an understanding of why the study and knowledge of algorithms are so important is to define exactly what we mean by an algorithm. An algorithm is any well-defined computational procedure that takes some value, or set of values, as input and produces some value, or set of values as output. In other words, algorithms are like road maps for accomplishing a given, well-defined task. So, a chunk of code that calculates the terms of the Fibonacci sequence is an implementation of a particular algorithm. Even a simple function for adding two numbers is an algorithm in a sense, albeit a simple one.

Some algorithms, like those that compute the Fibonacci sequences, are intuitive and may be innately embedded into our logical thinking and problem solving skills. However, for most of us, complex algorithms are best studied so we can use them as building blocks for more efficient logical problem solving in the future. In fact, you may be surprised to learn just how many complex algorithms people use every day when they check their e-mail or listen to music on their computers. This article will introduce some basic ideas related to the analysis of algorithms, and then put these into practice with a few examples illustrating why it is important to know about algorithms.

**Course Outcomes:**

After successful completion of this course, student will be able to:

1. comprehend notion of algorithmic complexity and logic of fundamental algorithms.
2. apply fundamental algorithms in real life problem solving.
3. identify and evaluate suitable data structures to solve a problem effectively and efficiently.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Programme Outcomes:** | **PO1** | **PO2** | **PO3** | **PO4** | **PO5** | **PO6** | **PO7** | **PO8** | **PO9** |
| **CO1 :**comprehend notion of algorithmic complexity and logic of fundamental algorithms. | √ |  |  |  | √ |  |  |  |  |
| **CO2:**apply fundamental algorithms in real life problem solving. | √ | √ | √ | √ | √ |  |  |  |  |
| **CO3:**identify and evaluate suitable data structure to solve a problem effectively and efficiently | √ |  | √ | √ |  |  |  |  |  |

**Syllabus:**

**Elementary Algorithmic:** Efficiency of Algorithms, Average & worst-case analysis, Elementary Operation

**Analysis Techniques:** Empirical, mathematical, Asymptotic analysis and related unconditional and conditional notations**.**

**Analysis of Algorithms:** Analyzing control structures: sequencing, “For” loops, Recursive calls, “While” and “repeat” loops, Amortized analysis

**Solving Recurrences:** Intelligent guesswork, Homogeneous recurrences, Inhomogeneous Recurrences, Change of variable, Range transformations, Master Theorem, Recurrence Tree

**Data Structures:** Heaps, Binomial heaps, Disjoint set structures

**Greedy Algorithms:** Graphs: Minimum spanning trees-Kruskal’s algorithm, Prim’s algorithm, Graphs: Shortest paths

**Divide-and-Conquer:** Multiplying large integers, Binary search, sorting: sorting by merging, quick sort, finding the median, Matrix multiplication, Exponentiation, approaches using recursion, memory functions.

**Dynamic Programming:** The principle of optimality, Various applications using Dynamic Programming.

**Branch and Bound, Backtracking**: Design of some classical problems using branch and bound and Backtracking approaches.

**Randomized and Approximation Algorithms:** Design of some classical problems**.**

**Self-Study:**

The self-study contents will be declared at the commencement of semester. Around 10% of the questions will be asked from self-study contents.

**References:**

* 1. Charles E. Leiserson, Thomas H. Cormen, Ronald L. Rivest, Clifford Stein - Introduction to Algorithms, PHI
  2. Gilles Brassard & Paul Bratley, Fundamentals of Algorithmic, PHI.
  3. Ellis Horowitz, Sartaj Sahni, Sanguthevar Rajasekharan, Fundamentals of Computer Algorithms, Galgotia.

**Tutorial details: ( problem sheet, schedule, assessment policy)**

**Component wise Continuous Evaluation & Semester End Examination weightage:**

**Lesson Plan**

|  |  |  |
| --- | --- | --- |
| **Lecture No.** | **Topic** | **Mapped CLO** |
| 1 | Introduction, Preliminaries, Representations of Algorithms, Efficiency of Algorithms, Average & worst-case analysis | 1 |
| 2 | Asymptotic Notation: A notation for “the order of”, other asymptotic notation | 1 |
| 3 | Examples based on Asymptotic Notations | 2 |
| 4 | Analysing control structures: sequencing, “For” loops, Recursive calls, “While” and “repeat” loops, using a barometer | 2 |
| 5 | Amortized analysis | 1,2 |
| 6 | Solving Recurrences: Intelligent guesswork. | 2 |
| 7 | Homogeneous Recurrences. | 1 |
| 8 | Examples solving of Homogeneous Recurrence | 2 |
| 9 | Inhomogeneous Recurrences | 2 |
| 10 | Examples solving of Inhomogeneous Recurrence | 2 |
| 11 | Change of variable method | 2 |
| 12 | Range transformations method | 2 |
| 13 | Master Method | 2 |
| 14 | Examples on Recurrence Tree | 2 |
| 15 | Disjoint Data Structure | 3 |
| 16 | Binary Heap and Binomial Heap | 3 |
| 17 | Divide and Conquer: Quick sort | 2,3 |
| 18 | Merge Sort | 2,3 |
| 19 | Finding Median | 2,3 |
| 20 | Strassen's Matrix Multiplication | 2,3 |
| 21 | Greedy Algorithms: Introduction, Minimum Spanning Tree: Prim's and Kruskal's algorithm | 3 |
| 22 | Scheduling: minimizing time in the system, Scheduling with dead line | 2,3 |
| 23 | Dynamic Programing: The principle of Optimality | 2 |
| 24 | Problems on Dynamic Programing: Making Change, Knapsack Problem | 2,3 |
| 25 | Problems on Dynamic Programing: Shortest Path, Chain matrix Multiplication | 2,3 |
| 26 | Backtracking:Examples on Backtracking : N Queen,  :Knapsack | 3 |
| 27 | Hamiltonian Cycles and Graph colouring | 3 |
| 28 | Branch and Bound and its examples | 3 |
| 29 | Introduction to Approximation Algorithm | 1,2 |
| 30 | NP Theory | 1,2 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Tutorial Plan**

|  |  |  |
| --- | --- | --- |
| **Tutorial No** | **Topic** | **Mapped CLO** |
| 1 | Basics of Algorithms | 1 |
| 2 | Asymptotic Notations | 1 & 2 |
| 3 | Recurrences | 3 |
| 4 | Disjoint Data Structures | 1 |
| 5 | Heap, Binomial Heap | 1 & 3 |
| 6 | Divide and Conquer | 2 |
| 7 | Greedy Algorithms | 2 |
| 8 | Dynamic Programing | 3 |
| 9 | Branch and Bound , Backtracking | 2,3 |
| 10 | Randomized and Approximation algorithms | 1 |

**List of Practicals**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No.** | **Title** | **Schedule (in Weeks)** | **CO** |
| 1. | To implement iterative and full recursive version of following sorting algorithms: Selection Sort, Insertion Sort and Bubble Sort. | 1 | CO1,2 |
| 2. | To implement randomized quick sort. | 2 | CO2,3 |
| 3. | To implement merge sort under certain constraints of file size and copying the sorted data in another file. | 3,4 | CO2,3 |
| 4. | To implement Maximum subarray sum problem. | 5,6 | CO2,3 |
| 5. | Implementation of problem based on Greedy aproach | 7 | CO1,3 |
| 6. | Implementation of problem based on Greedy approach | 8 | CO1,3 |
| 7. | Implementation of problem based on Dynamic Programming | 9,10 | CO1,3 |
| 8. | Implementation of problem based on Dynamic Programming | 11,12 | CO1,3 |
| 9. | Implementation of problem based on Dynamic Programming | 13 | CO1,3 |
| 10. | Implementation of problem based on Backtracking and Branch-and-Bound | 14,15 | CO1,3 |

**Course Assessment Schemes**

**(Course without Laboratory & Tutorial components)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Assessment scheme** | **CE** | | | **SEE** |
| **Component weightage** | 0.6 | | | 0.4 |
|  | Class Test  30% | Sessional Exam  40% | Tutorial Evaluation  30% |  |

**Teaching-learning methodology:**

* Lectures: Blackboard , PPT, Discussion
* Tutorial: Innovative ways of Numerical solving, derivations, Problem Solving, Application of Mathematical Models to real Systems etc.

**Active learning techniques**

* Flipped Class-room (Topics to be mentioned) , Muddiest Points
* Others (Specify)

**Types of Special/Innovative Assignments, Term Papers, mini Projects etc.**

* Assignments and Quizzes

**Course Material:**

* Course Policy
* PPTs, Notes, other Material:

<https://sites.google.com/a/nirmauni.ac.in/ce601-design-and-analysis-of-algorithms-2016-17/home/academic-docs/2009-10-odd/h-lecture-notes>

* [Assignments, Tutorials](https://sites.google.com/a/nirmauni.ac.in/3ec1218-testing-and-verification-of-vlsi-design/home/academic-docs/j-assignments-tutorials), [Lab Manuals](https://sites.google.com/a/nirmauni.ac.in/3ec1218-testing-and-verification-of-vlsi-design/home/academic-docs/l-laboratory-manuals) :

<https://sites.google.com/a/nirmauni.ac.in/ce601-design-and-analysis-of-algorithms-2016-17/home/academic-docs/2009-10-odd/j-assignments-tutorials>

* Question bank:

<https://sites.google.com/a/nirmauni.ac.in/ce601-design-and-analysis-of-algorithms-2016-17/home/academic-docs/2009-10-odd/v-question-bank-optional>

* [Web-links](https://sites.google.com/a/nirmauni.ac.in/3ec1218-testing-and-verification-of-vlsi-design/home/academic-docs/m-course-related-important-web-links), [Blogs](https://sites.google.com/a/nirmauni.ac.in/3ec1218-testing-and-verification-of-vlsi-design/home/academic-docs/n-course-related-own-blog-and-other-such-blogs), [Video Lectures](https://sites.google.com/a/nirmauni.ac.in/3ec1218-testing-and-verification-of-vlsi-design/home/academic-docs/o-video-lectures-if-available-like-nptel-mooc-etc), [Journals](https://sites.google.com/a/nirmauni.ac.in/3ec1218-testing-and-verification-of-vlsi-design/home/academic-docs/p-list-of-international-national-journals-related-to-the-course)  :

<https://sites.google.com/a/nirmauni.ac.in/ce601-design-and-analysis-of-algorithms-2016-17/home/academic-docs/2009-10-odd/o-video-lectures-if-available-like-nptel-mooc-etc>

* [Animations /Simulations](https://sites.google.com/a/nirmauni.ac.in/3ec1218-testing-and-verification-of-vlsi-design/home/academic-docs/x-animations-simulations-to-explain-the-complex-principles), [Softwares](https://sites.google.com/a/nirmauni.ac.in/3ec1218-testing-and-verification-of-vlsi-design/home/academic-docs/y-useful-softwares)
* [Advanced topics](https://sites.google.com/a/nirmauni.ac.in/3ec1218-testing-and-verification-of-vlsi-design/home/academic-docs/q-list-of-advanced-topics-seminar-topics-related-to-the-course)
* Industries/[Organizations](https://sites.google.com/a/nirmauni.ac.in/3ec1218-testing-and-verification-of-vlsi-design/home/academic-docs/r-list-of-world-leading-industries-organizations-working-on-the-course-related-areas)

**Course Outcome Attainment:**

* Use of formal evaluation components of continuous evaluation, tutorials, laboratory work, semester end examination
* Informal feedback during course conduction
* Surveys & Peer observation