CS F222: Discrete Structures for Computer Science

# Course Project

An Application of Graph Optimisation
University Course Assignment System

By

Anirudh Sharma - 2022A7PS0657G

Arnav Mangla - 2022A7PS0244G

Priya Rathi - 2022A7PS1096G

Department of Computer Science and Information Systems

Birla Institute of Technology and Science Pilani, KK Birla Goa Campus

# Glossary

1.FD CDC: First Degree Compulsory Discipline Course

2.FD EL: First Degree (Discipline) Elective

3.HD CDC: Higher Degree Compulsory Discipline Course

4.HD EL: Higher Degree (Discipline) Elective

5.CS: Computer Science

6.S: Source

7.T: Sink

# Question

The research problem at hand revolves around optimising the University Course Assignment System. Within a department, there are n faculty members categorised into three distinct groups: x1, x2, and x3. Faculty in each category are assigned different course loads, with x1 handling 0.5 courses per semester, x2 taking one course per semester, and x3 managing 1.5 courses per semester.

In this system, faculty members have the flexibility to take multiple courses in a given semester, and conversely, a single course can be assigned to multiple faculty members. When a course is shared between two professors, each professor's load is considered to be 0.5 courses. Moreover, each faculty member maintains a preference list of courses, ordered by their personal preferences, with the most preferred courses appearing at the top. Importantly, there is no prioritisation among faculty members within the same category.

The primary objective of this research problem is to develop an assignment scheme that maximises the number of courses assigned to faculty while aligning with their preferences and the category-based constraints (x1, x2, x3). The challenge lies in ensuring that a course can only be assigned to a faculty member if it is on their preference list.

This problem is unique due to its flexibility regarding the number of courses faculty members can take, distinct from typical assignment problems. Potential modifications may include adjusting the maximum number of courses "y" for each category of professors instead of requiring exact adherence or extending the number of professor categories beyond the existing three to devise a more generalised solution.

## Interpretation and Assumptions

1.The preference list has been considered on an opt-in/opt-out basis. All courses in the preference list of a professor have been given equal priority.

2.Each faculty is assigned course(s) present only in their preference list. They can only be assigned a course in it.

3.This problem has been solved considering only the CS Dept. of BITS Pilani KK Birla Goa Campus. Hence, only CS Courses and Professors have been accounted for, 29 and 30 in number, respectively.

4.Each professor has filled at least 4 FD CDCs, 4 FD ELs, 2 HD CDCs, and 2 HD ELs in their preference list.

5.Each professor has been assigned different course loads: x1 handles 0.5 courses per semester, x2 handles one course per semester, and x3 handles 1.5 courses per semester.

6.The plan was to consider the courses for the First Semester of an academic year; however, since there are no CS CDCs in the first semester and none in the seventh semester, we have considered 29 Courses: 12 FDCDCs, 6 FDELs, 5 HDCDCs and 6HDELs.

7.There is no prioritisation among professors within the same category.

8.As per our inputs, we have fixed the courses and the professors, and we will only be varying the preference lists of the professors.

9.An equal number of x1, x2, and x3 types of professors (= 10 each). And the number of courses almost matches the maximum that this collection of professors can have.

# Method Approach

After carefully reviewing the problem statement, we decided to use C++ as our programming language of choice. We quickly realised that a Bipartite-directed graph would be the best approach, with one set of vertices representing professors and the other set representing courses. In this graph, an edge between a professor and a course represents an assignment.

Firstly, we made text files for input: “Courses.txt” is for courses with their codes, “Profs.txt” is for professors with their codes and their categories, and “Prof_plist1/2/2.txt” is for each professor’s preference list.

Then, we need to store all of this data in our program. The data structure we chose to use is classes: Professor and Course. Each professor/course object holds all information for that professor/course. We built vectors storing codes of professors and courses and maps that go from a professor’s/course’s code to the pointer of the respective object.

The graph we have used is a 2D vector with nodes for professors, courses, a sink node, and a source node, as we used the concept of network flow to solve this problem, such that we want “max flow and min cut.”

To simplify the graph slightly, we have made it an unweighted bipartite graph, and we will try to maximise the bipartite matching using a combination of Breadth-First Search (BFS) and the Ford Fulkerson Algorithm.

1. Breadth-first search is an algorithm for searching a graph/tree data structure for a node that satisfies a given property. It starts at a node and explores all adjacent nodes before moving on to the nodes at the next depth level.

2. Ford-Fulkerson Algorithm - The algorithm works by iteratively finding an augmenting path, which is a path from the source to the sink in the residual graph, i.e., the graph obtained by subtracting the current flow from the capacity of each edge. The algorithm then increases the flow along this path by the maximum possible amount, the minimum capacity of the edges. S and T nodes represent the beginning and end of the flow network.

Each professor is classified into three categories: x1, x2, and x3. x1 will be denoted by one, x2 by two and x3 by three, representing the maximum number of courses a professor can take.

A professor node has been defined as three separate nodes, and a course node has been defined as two different nodes. This has been so that after running our algorithm, each node has only one node adjacent to it.

If the category of a professor is ‘i’, only the first ‘i’ of the three nodes will have any edges adjacent to it.

For this reason, we defined our vector with dimensions = n x n, where

n = 3x(No. Of Professors) + 2x(No. Of Courses) + 1 (sink node) + 1 (source node)

In our approach, all edges from the source node are directed to the professor nodes. The professor nodes are directed to the course nodes, and the course nodes are directed to the sink node. Initially, we mark all edges corresponding to the courses in the professor’s preference list.

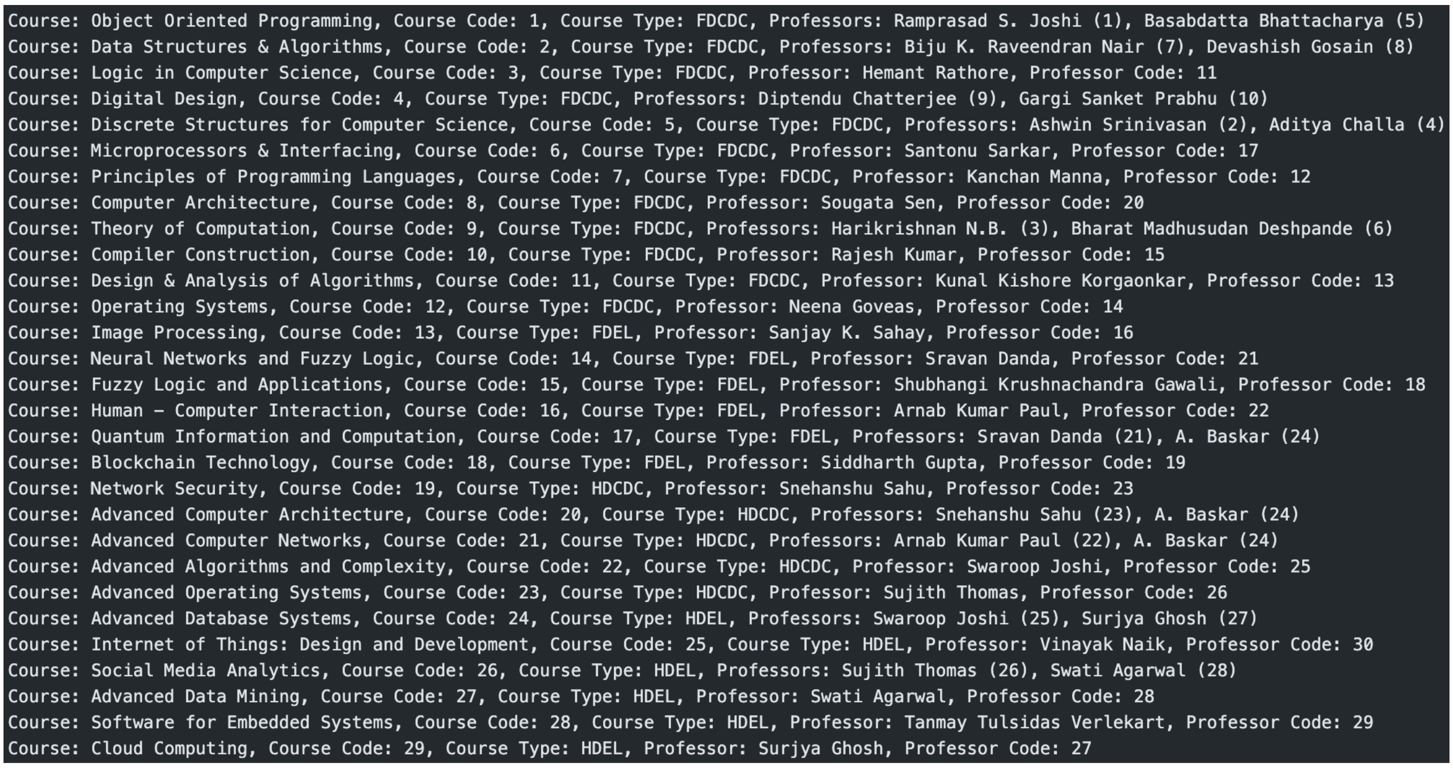
A print function was written to print our initial graph in a text file “original_graph.txt”. This graph is based on the preference list of each professor. As stated earlier, we output our graph as a matrix. Each graph cell consists of either a 0 or 1 value. One represents the presence of the course in the professor’s preference list, whereas 0 represents the absence of the course in the professor’s preference list.

Then, the Ford-Fulkerson algorithm runs and updates the residual graph (rGraph) and simultaneously creates a new graph (newGraph) that stores all the correct assignments. The graph of the final assignments is printed in a text file “Output.txt”.

A function: printAssignment was written to present the graph readably in a file “assignment.txt”. This stores the course code, course name, and the corresponding professor(s)’ name(s) and their codes. If a valid assignment could not be produced for that course, it was denoted by “[COULD NOT BE ASSIGNED]”. If it was only assigned partially, then it was suggested by “[COURSE ALLOTTED PARTIALLY].”

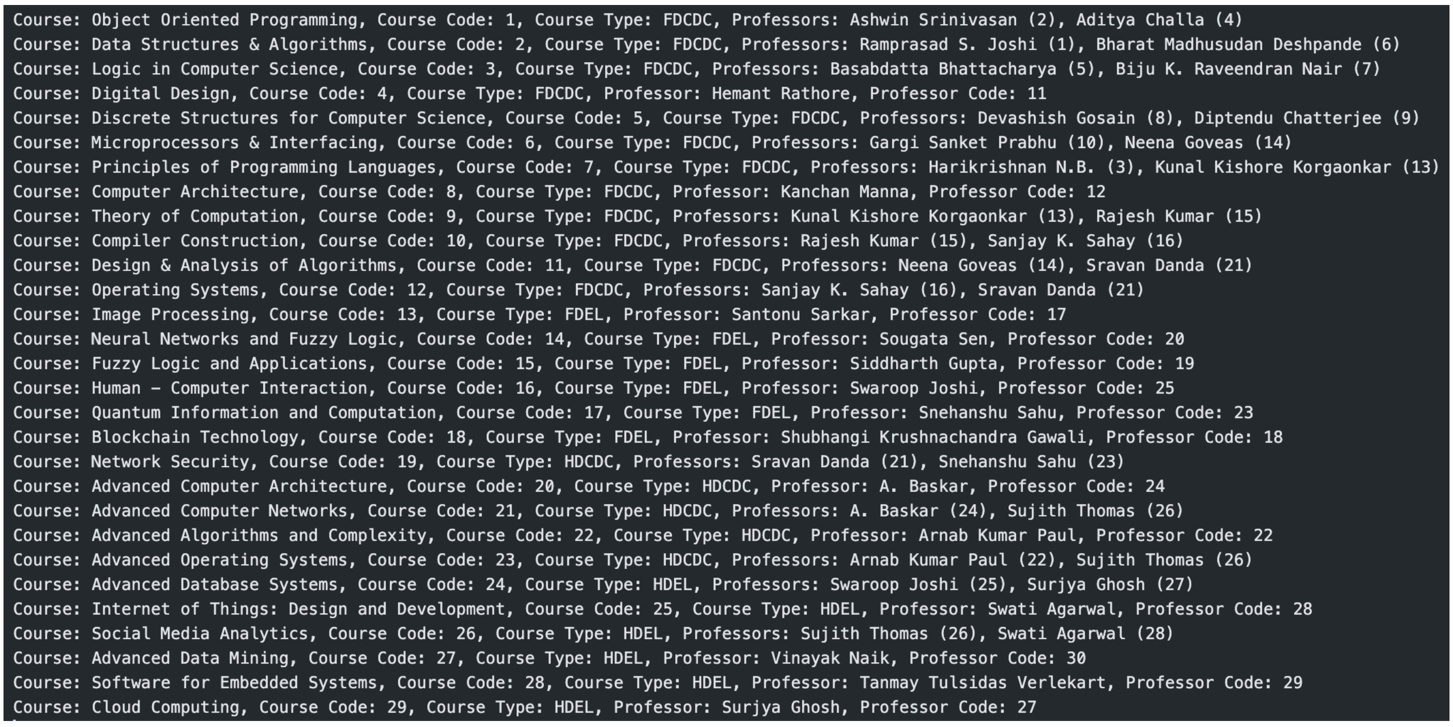
# Results

Prof_plist1.txt:



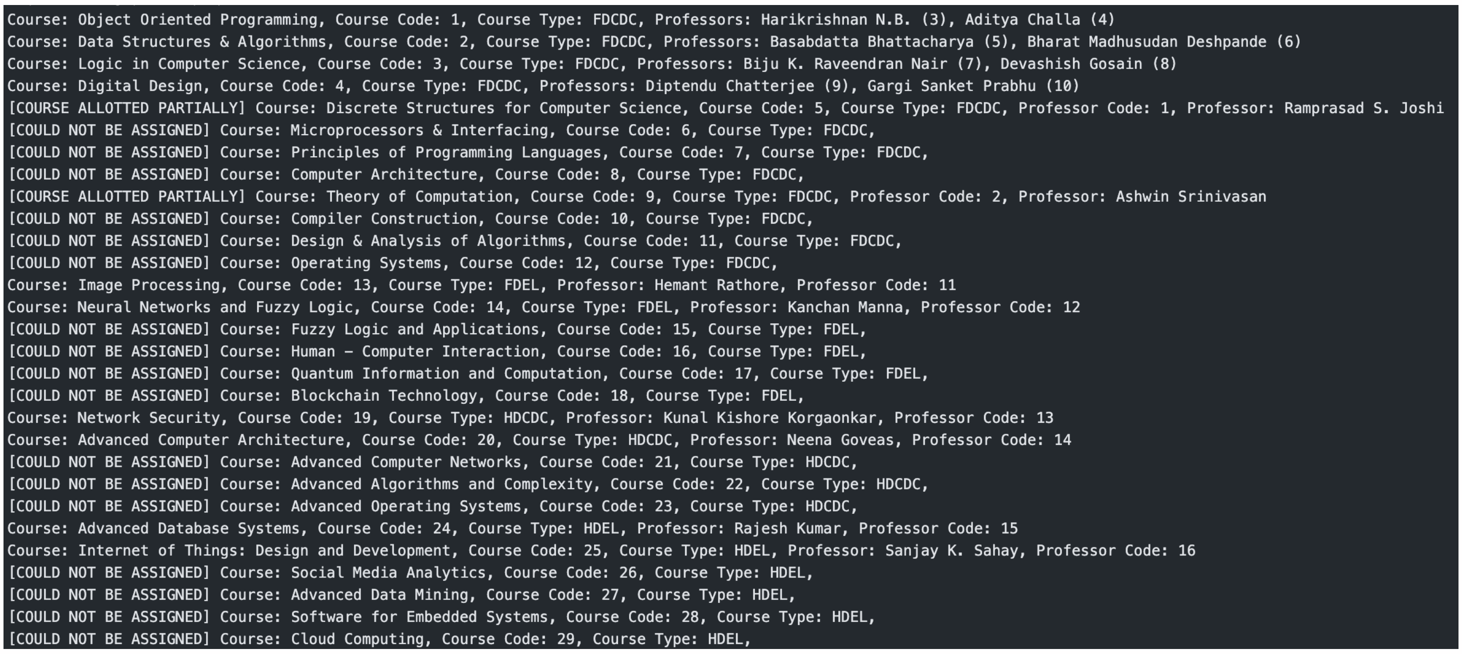
```
Course: Object Oriented Programming, Course Code: 1, Course Type: FDCDC, Professors: Ramprasad S. Joshi (1), Basabdatta Bhattacharya (5)
Course: Data Structures & Algorithms, Course Code: 2, Course Type: FDCDC, Professors: Biju K. Raveendran Nair (7), Devashish Gosain (8)
Course: Logic in Computer Science, Course Code: 3, Course Type: FDCDC, Professor: Hemant Rathore, Professor Code: 11
Course: Digital Design, Course Code: 4, Course Type: FDCDC, Professors: Diptendu Chatterjee (9), Gargi Sanket Prabhu (10)
Course: Discrete Structures for Computer Science, Course Code: 5, Course Type: FDCDC, Professors: Ashwin Srinivasan (2), Aditya Challa (4)
Course: Microprocessors & Interfacing, Course Code: 6, Course Type: FDCDC, Professor: Santonu Sarkar, Professor Code: 17
Course: Principles of Programming Languages, Course Code: 7, Course Type: FDCDC, Professor: Kanchan Manna, Professor Code: 12
Course: Computer Architecture, Course Code: 8, Course Type: FDCDC, Professor: Sougata Sen, Professor Code: 20
Course: Theory of Computation, Course Code: 9, Course Type: FDCDC, Professors: Harikrishnan N.B. (3), Bharat Madhusudan Deshpande (6)
Course: Compiler Construction, Course Code: 10, Course Type: FDCDC, Professor: Rajesh Kumar, Professor Code: 15
Course: Design & Analysis of Algorithms, Course Code: 11, Course Type: FDCDC, Professor: Kunal Kishore Korgaonkar, Professor Code: 13
Course: Operating Systems, Course Code: 12, Course Type: FDCDC, Professor: Neena Goveas, Professor Code: 14
Course: Image Processing, Course Code: 13, Course Type: FDEL, Professor: Sanjay K. Sahay, Professor Code: 16
Course: Neural Networks and Fuzzy Logic, Course Code: 14, Course Type: FDEL, Professor: Sravan Danda, Professor Code: 21
Course: Fuzzy Logic and Applications, Course Code: 15, Course Type: FDEL, Professor: Shubhangi Krushnachandra Gawali, Professor Code: 18
Course: Human - Computer Interaction, Course Code: 16, Course Type: FDEL, Professor: Arnab Kumar Paul, Professor Code: 22
Course: Quantum Information and Computation, Course Code: 17, Course Type: FDEL, Professors: Sravan Danda (21), A. Baskar (24)
Course: Blockchain Technology, Course Code: 18, Course Type: FDEL, Professor: Siddharth Gupta, Professor Code: 19
Course: Network Security, Course Code: 19, Course Type: HDCDC, Professor: Snehanshu Sahu, Professor Code: 23
Course: Advanced Computer Architecture, Course Code: 20, Course Type: HDCDC, Professors: Snehanshu Sahu (23), A. Baskar (24)
Course: Advanced Computer Networks, Course Code: 21, Course Type: HDCDC, Professors: Arnab Kumar Paul (22), A. Baskar (24)
Course: Advanced Algorithms and Complexity, Course Code: 22, Course Type: HDCDC, Professor: Swaroop Joshi, Professor Code: 25
Course: Advanced Operating Systems, Course Code: 23, Course Type: HDCDC, Professor: Sujith Thomas, Professor Code: 26
Course: Advanced Database Systems, Course Code: 24, Course Type: HDEL, Professors: Swaroop Joshi (25), Surjya Ghosh (27)
Course: Internet of Things: Design and Development, Course Code: 25, Course Type: HDEL, Professor: Vinayak Naik, Professor Code: 30
Course: Social Media Analytics, Course Code: 26, Course Type: HDEL, Professors: Sujith Thomas (26), Swati Agarwal (28)
Course: Advanced Data Mining, Course Code: 27, Course Type: HDEL, Professor: Swati Agarwal, Professor Code: 28
Course: Software for Embedded Systems, Course Code: 28, Course Type: HDEL, Professor: Tanmay Tulsidas Verlekart, Professor Code: 29
Course: Cloud Computing, Course Code: 29, Course Type: HDEL, Professor: Surjya Ghosh, Professor Code: 27
```

Prof_plist2.txt:



```
Course: Object Oriented Programming, Course Code: 1, Course Type: FDCDC, Professors: Ashwin Srinivasan (2), Aditya Challa (4)
Course: Data Structures & Algorithms, Course Code: 2, Course Type: FDCDC, Professors: Ramprasad S. Joshi (1), Bharat Madhusudan Deshpande (6)
Course: Logic in Computer Science, Course Code: 3, Course Type: FDCDC, Professors: Basabdatta Bhattacharya (5), Biju K. Raveendran Nair (7)
Course: Digital Design, Course Code: 4, Course Type: FDCDC, Professor: Hemant Rathore, Professor Code: 11
Course: Discrete Structures for Computer Science, Course Code: 5, Course Type: FDCDC, Professors: Devashish Gosain (8), Diptendu Chatterjee (9)
Course: Microprocessors & Interfacing, Course Code: 6, Course Type: FDCDC, Professors: Gargi Sanket Prabhu (10), Neena Goveas (14)
Course: Principles of Programming Languages, Course Code: 7, Course Type: FDCDC, Professors: Harikrishnan N.B. (3), Kunal Kishore Korgaonkar (13)
Course: Computer Architecture, Course Code: 8, Course Type: FDCDC, Professor: Kanchan Manna, Professor Code: 12
Course: Theory of Computation, Course Code: 9, Course Type: FDCDC, Professors: Kunal Kishore Korgaonkar (13), Rajesh Kumar (15)
Course: Compiler Construction, Course Code: 10, Course Type: FDCDC, Professors: Rajesh Kumar (15), Sanjay K. Sahay (16)
Course: Design & Analysis of Algorithms, Course Code: 11, Course Type: FDCDC, Professors: Neena Goveas (14), Sravan Danda (21)
Course: Operating Systems, Course Code: 12, Course Type: FDCDC, Professors: Sanjay K. Sahay (16), Sravan Danda (21)
Course: Image Processing, Course Code: 13, Course Type: FDEL, Professor: Santonu Sarkar, Professor Code: 17
Course: Neural Networks and Fuzzy Logic, Course Code: 14, Course Type: FDEL, Professor: Sougata Sen, Professor Code: 20
Course: Fuzzy Logic and Applications, Course Code: 15, Course Type: FDEL, Professor: Siddharth Gupta, Professor Code: 19
Course: Human - Computer Interaction, Course Code: 16, Course Type: FDEL, Professor: Swaroop Joshi, Professor Code: 25
Course: Quantum Information and Computation, Course Code: 17, Course Type: FDEL, Professor: Snehanshu Sahu, Professor Code: 23
Course: Blockchain Technology, Course Code: 18, Course Type: FDEL, Professor: Shubhangi Krushnachandra Gawali, Professor Code: 18
Course: Network Security, Course Code: 19, Course Type: HDCDC, Professors: Sravan Danda (21), Snehanshu Sahu (23)
Course: Advanced Computer Architecture, Course Code: 20, Course Type: HDCDC, Professor: A. Baskar, Professor Code: 24
Course: Advanced Computer Networks, Course Code: 21, Course Type: HDCDC, Professors: A. Baskar (24), Sujith Thomas (26)
Course: Advanced Algorithms and Complexity, Course Code: 22, Course Type: HDCDC, Professor: Arnab Kumar Paul, Professor Code: 22
Course: Advanced Operating Systems, Course Code: 23, Course Type: HDCDC, Professors: Arnab Kumar Paul (22), Sujith Thomas (26)
Course: Advanced Database Systems, Course Code: 24, Course Type: HDEL, Professors: Swaroop Joshi (25), Surjya Ghosh (27)
Course: Internet of Things: Design and Development, Course Code: 25, Course Type: HDEL, Professor: Swati Agarwal, Professor Code: 28
Course: Social Media Analytics, Course Code: 26, Course Type: HDEL, Professors: Sujith Thomas (26), Swati Agarwal (28)
Course: Advanced Data Mining, Course Code: 27, Course Type: HDEL, Professor: Vinayak Naik, Professor Code: 30
Course: Software for Embedded Systems, Course Code: 28, Course Type: HDEL, Professor: Tanmay Tulsidas Verlekart, Professor Code: 29
Course: Cloud Computing, Course Code: 29, Course Type: HDEL, Professor: Surjya Ghosh, Professor Code: 27
```

Prof_plist3.txt:



```
Course: Object Oriented Programming, Course Code: 1, Course Type: FDCDC, Professors: Harikrishnan N.B. (3), Aditya Challa (4)
Course: Data Structures & Algorithms, Course Code: 2, Course Type: FDCDC, Professors: Basabdatta Bhattacharya (5), Bharat Madhusudan Deshpande (6)
Course: Logic in Computer Science, Course Code: 3, Course Type: FDCDC, Professors: Biju K. Raveendran Nair (7), Devashish Gosain (8)
Course: Digital Design, Course Code: 4, Course Type: FDCDC, Professors: Diptendu Chatterjee (9), Gargi Sanket Prabhu (10)
[COURSE ALLOTTED PARTIALLY] Course: Discrete Structures for Computer Science, Course Code: 5, Course Type: FDCDC, Professor Code: 1, Professor: Ramprasad S. Joshi
[COULD NOT BE ASSIGNED] Course: Microprocessors & Interfacing, Course Code: 6, Course Type: FDCDC,
[COULD NOT BE ASSIGNED] Course: Principles of Programming Languages, Course Code: 7, Course Type: FDCDC,
[COULD NOT BE ASSIGNED] Course: Computer Architecture, Course Code: 8, Course Type: FDCDC,
[COURSE ALLOTTED PARTIALLY] Course: Theory of Computation, Course Code: 9, Course Type: FDCDC, Professor Code: 2, Professor: Ashwin Srinivasan
[COULD NOT BE ASSIGNED] Course: Compiler Construction, Course Code: 10, Course Type: FDCDC,
[COULD NOT BE ASSIGNED] Course: Design & Analysis of Algorithms, Course Code: 11, Course Type: FDCDC,
[COULD NOT BE ASSIGNED] Course: Operating Systems, Course Code: 12, Course Type: FDCDC,
Course: Image Processing, Course Code: 13, Course Type: FDEL, Professor: Hemant Rathore, Professor Code: 11
Course: Neural Networks and Fuzzy Logic, Course Code: 14, Course Type: FDEL, Professor: Kanchan Manna, Professor Code: 12
[COULD NOT BE ASSIGNED] Course: Fuzzy Logic and Applications, Course Code: 15, Course Type: FDEL,
[COULD NOT BE ASSIGNED] Course: Human - Computer Interaction, Course Code: 16, Course Type: FDEL,
[COULD NOT BE ASSIGNED] Course: Quantum Information and Computation, Course Code: 17, Course Type: FDEL,
[COULD NOT BE ASSIGNED] Course: Blockchain Technology, Course Code: 18, Course Type: FDEL,
Course: Network Security, Course Code: 19, Course Type: HDCDC, Professor: Kunal Kishore Korgaonkar, Professor Code: 13
Course: Advanced Computer Architecture, Course Code: 20, Course Type: HDCDC, Professor: Neena Goveas, Professor Code: 14
[COULD NOT BE ASSIGNED] Course: Advanced Computer Networks, Course Code: 21, Course Type: HDCDC,
[COULD NOT BE ASSIGNED] Course: Advanced Algorithms and Complexity, Course Code: 22, Course Type: HDCDC,
[COULD NOT BE ASSIGNED] Course: Advanced Operating Systems, Course Code: 23, Course Type: HDCDC,
Course: Advanced Database Systems, Course Code: 24, Course Type: HDEL, Professor: Rajesh Kumar, Professor Code: 15
Course: Internet of Things: Design and Development, Course Code: 25, Course Type: HDEL, Professor: Sanjay K. Sahay, Professor Code: 16
[COULD NOT BE ASSIGNED] Course: Social Media Analytics, Course Code: 26, Course Type: HDEL,
[COULD NOT BE ASSIGNED] Course: Advanced Data Mining, Course Code: 27, Course Type: HDEL,
[COULD NOT BE ASSIGNED] Course: Software for Embedded Systems, Course Code: 28, Course Type: HDEL,
[COULD NOT BE ASSIGNED] Course: Cloud Computing, Course Code: 29, Course Type: HDEL,
```

All of these outputs have been verified to be correct by checking with the respective plist files and checking all the constraints provided