	School of Engineering & Technology	
	Department: SOET	**Session: 2025-26**
	Program: BCA (AI&DS)	**Semester: V**
	Course Code: ENCA351	**Number of students: 188**
	Course Name: Design and Analysis of Algorithms Lab	**Faculty: Dr. Aarti**

Lab Assignment -4: Airline Crew Scheduling – NP-Hard Problem Solving

Instructions:

- Assignment must be submitted within the deadline communicated by the instructor at the time of release.
- Assignment must be submitted on https://lms.krmangalam.edu.in/
- You must provide a link to your GitHub repository with your submission on LMS.
- Use of ChatGPT and similar tools is strictly prohibited.
- The assignment needs to be submitted by each individual.
- This assignment carries a total of 10 marks.
- Assignments will be assessed based on the evaluation rubrics.
- Estimated Duration: 6-8 hours

Assignment Title: Solving Airline Crew Scheduling Using Backtracking and Constraint Satisfaction

---

Real-World Problem Context

Scheduling airline crew members efficiently is a classic example of an NP-hard problem in resource allocation. It involves assigning flights to crew members under complex constraints such as availability, rest periods between flights, non-overlapping schedules, and optional cost optimization. This mini project allows students to model and solve a simplified version of this real-world problem using a backtracking algorithm, understanding its computational complexity and practical implications.

Learning Objectives

By completing this project, you will:

- Understand how real-world scheduling problems can be modeled as NP-hard problems.
- Apply constraint satisfaction and backtracking to solve small-scale scheduling problems.
- Analyze and discuss the performance limitations of exhaustive search methods.
- Profile and visualize algorithm behavior in constrained environments.
- Communicate solution strategies and trade-offs effectively in technical documentation.

Assignment Tasks

Problem: Airline Crew Scheduling

- **Algorithmic Strategy**: Backtracking (Constraint Satisfaction)
- **Application Domain**: Airline Resource Allocation
- **Problem Description**: Assign a set of flights to available crew members while satisfying the following constraints:
    - No overlapping flights assigned to the same crew member.
    - Minimum rest time (e.g., 1 hour) between two flights assigned to the same crew.
    - (Optional) Minimize the total cost or maximize fairness of assignments.

Sub-Tasks:

1. Input

- Create a list of flights in the form:

```
flights = [('F1', 9, 11), ('F2', 10, 12), ('F3', 13, 15), ...]
```

- Each tuple contains: (Flight ID, Start Time, End Time)
- Create a list of available crew members:

```
crew_members = ['C1', 'C2', 'C3']
```

---

2. Approach

- Write a **constraint checker** function to ensure no flight overlaps and minimum rest time between flights is satisfied.

- Generate **all valid non-overlapping flight combinations** for each crew member.
- Implement **recursive backtracking** to assign flights to crew members. If at any point no valid assignment exists, backtrack and try another combination.
- (Optional) Add **cost values** to each flight-crew pair and attempt to minimize total cost.

---

3. Output

- A dictionary or mapping from each crew member to the list of flights assigned. Example:

```
{
    'C1': ['F1', 'F4'],
    'C2': ['F2', 'F5'],
    'C3': ['F3']
}
```

---

4. Analysis

- Discuss the **NP-hard nature** of the problem.
- Explain why brute-force and backtracking work only for small n.
- Provide the **time complexity** (Exponential: $O(k \times 2^n)$, where n = flights, k = crew).
- Suggest real-world improvements: use of heuristics, integer programming, or optimization libraries.

---

5. Visualization

- (Optional) Create a **Gantt chart** to visualize flight schedules across crew members using matplotlib.
- Plot **execution time vs. number of flights** to show how quickly complexity grows.

Task 2: Experimental Profiling & Visualization

- Use time and memory_profiler to measure performance for different input sizes.
- Plot time taken for increasing number of flights (e.g., 4 to 10).

- Discuss where and why backtracking becomes infeasible.
- (Optional) Track number of recursive calls for deeper insights.

Task 3: Final Summary and Documentation

Include the following in your submission:

Element	Description
Problem	Airline Crew Scheduling
Strategy	Backtracking (Constraint Satisfaction)
Time Complexity	$O(k \times 2^n)$
Domain	Resource Allocation in Aviation
Notes	Infeasible for large datasets; requires optimization techniques

- Summary of insights: strengths, limitations, practical relevance
- README.md with problem explanation, setup steps, and citations
- Annotated code and markdown explanations in the notebook

Evaluation Rubric (Total: 10 Marks)

Criteria	Marks	Description
GitHub Setup & Organization	1	Clean repo with README, .gitignore, and environment setup
Algorithm Implementation	3	Accurate backtracking logic, well-commented code
Profiling & Visualization	2.5	Use of profiling tools and basic input scaling analysis
Analysis & Insights	2.5	Clarity on problem complexity, limitations, and improvements
Code Quality & Documentation	1	Markdown use, comments, clean output formatting

---

Submission Instructions

➢ Push the following to your GitHub repository:

- README.md: project overview and setup
- crew_scheduling_notebook.ipynb: all code and plots
- images/ (optional): Gantt charts or output screenshots
- requirements.txt: package list
- .gitignore and venv files (optional)

- Submit the GitHub repo link via LMS.

Academic Integrity & Plagiarism Policy

- This is an individual assignment.
- Plagiarized code or reports will result in **zero marks**.
- Cite any references or tools used in README.md.

Support Resources

- *Introduction to Algorithms* – Cormen et al. (CLRS)
- Python libraries: itertools, time, memory_profiler, matplotlib
- Optional tools: ortools, PuLP, z3

Contact: Dr. Aarti, aarti.sangwan@krmangalam.edu.in