Assignment: Build a Student Campus Navigator -- "BotBrain"

Weight: 10%

Duration: 4 weeks

Submission Date: 17-09-2025

Teams: Individual (Full Marks) / Team of 2 (Partial marks deduction by 30%)

Deliverables: Code + design report + live/demo session + Submission in Github with detailed readme and code documentation

Objective

Design and implement an intelligent agent ("**BotBrain**") that helps students navigate the **Chanakya University campus** both digitally and physically. Your solution must demonstrate understanding of search algorithms, agent design, and problem-solving methodologies covered in the first 8 weeks of the course.

Core Requirements (80% of marks)

1. Campus Environment Modeling (20 marks)

- Create a campus representation with:
  - **Minimum 12 buildings/locations** including: Academic blocks (A, B, C), Library, Admin building, Main hostel, Canteen, Sports complex, Medical center, Main gate, Student center, Auditorium
  - **Simple path connections** with walking distances in meters
  - **Basic constraints**: Some paths may be one-way or have different walking speeds
  - **Requirement**: Provide a simple graph representation where each building is a node and paths are edges with distance weights. Draw your campus map clearly showing all connections.

2. Search Algorithm Implementation (30 marks)

- Implement the following search algorithms learned in class:
  - **Breadth-First Search (BFS)**
  - **Depth-First Search (DFS)**
  - **Uniform Cost Search (UCS)**
  - *A Search with Euclidean distance heuristic**

- **Requirements**:
  - Each algorithm should find paths between any two campus locations
  - Implement proper data structures (queues, stacks, priority queues)
  - Display the search process step-by-step showing nodes explored

3. Basic Query Processing and Information Services (20 marks)

Your system should handle simple navigation queries and provide basic information:

- "Find path from Hostel to Library"
- "Show route from Academic Block A to Canteen"
- "Navigate from Main Gate to Admin Block"

Features Required:

- Simple text-based interface for source and destination input
- Allow user to choose which search algorithm to use
- Display the found path, total distance, and estimated walking time
- Show basic building information (e.g., "Library: Study halls available, Open 8 AM - 10 PM")

4. Algorithm Comparison (10 marks)

Bloom's Level 4 (Analyze)

Compare the implemented algorithms by:

- Running each algorithm on 3 different source-destination pairs
- Recording number of nodes explored by each algorithm
- Creating a simple comparison table
- Writing a brief explanation of which algorithm found the shortest path

Additional Features (20% of marks)

5. Agent Concepts and Architecture (10 marks)

Bloom's Level 2 (Understand)

Explain your BotBrain as an intelligent agent by describing:

- **Agent Environment**: What is the environment your agent operates in?
- **Agent Type**: Is your agent a simple reflex agent or goal-based agent? Justify your answer
- **PEAS Analysis**: Define Performance measure, Environment, Actuators, and Sensors for your
- BotBrain **Agent Behavior**: How does your agent make decisions when finding paths?

6. System Enhancement (10 marks)

Bloom's Level 3 (Apply)

Choose ONE of the following enhancements:

- **Visual Output**: Create a simple graphical display of the campus map and highlight the path
- found

Building Information: Display relevant services/offices when showing paths (e.g., "Admin Block: Registrar office, Fee payment counter")

- **Path Alternatives**: Show the user 2 different path options with their respective distances
- **Simple Recommendations**: If destination is closed or unavailable, suggest nearby alternatives

Deliverables and Assessment

1. Code Implementation (40 marks)

Grading Criteria:

- **Excellent (A)**: All four search algorithms correctly implemented with proper data structures, code is well-organized
- **Good (B)**: Three algorithms work correctly with minor issues in implementation
- **Satisfactory (C)**: Two algorithms work with basic functionality
- **Poor (D)**: Only one algorithm works or significant implementation errors

2. Technical Report (30 marks)

Required Sections (5-7 pages):

1. **Agent Analysis and Campus Model** (10 marks)
    - PEAS analysis of your BotBrain agent
    - Description of campus graph representation
    - Simple justification of building connections
2. **Algorithm Implementation** (15 marks) Explanation of each search algorithm
    - Data structures used for implementation
    - Step-by-step example showing how each algorithm works
3. **Basic Comparison** (5 marks)
    - Simple results table comparing the algorithms
    - Brief explanation of which algorithm found the shortest path

3. Live Demonstration (20 marks)

10-minute presentation including:

- **System Overview** (3 minutes): Show campus model and explain basic agent design
- **Algorithm Demonstration** (5 minutes): Run each search algorithm on examples
- **Q&A** (2 minutes): Answer simple questions about implementation

4. Code Documentation & GitHub (10 marks)

- Clear README with basic usage instructions
- Code comments explaining algorithm logic
- Simple examples of input and output

Anti-AI/ChatGPT Measures

1. **Specific Campus Layout**: Must use actual Chanakya University building locations with justified connections
2. **Step-by-Step Algorithm Traces**: Show detailed execution traces for each algorithm on specific examples
3. **Personal Implementation Choices**: Explain why you chose specific data structures and implementation approaches
4. **Live Demonstration**: Real-time coding demonstration and answering questions about implementation details
5. **Handwritten Algorithm Traces**: Submit hand-drawn traces for at least one example per algorithm
6. **Original Analysis**: Personal insights about algorithm performance based on your specific implementation and testing

Timeline (4 Weeks)

Week	Tasks	Deliverable
1	Create campus map, design graph structure, start with BFS and DFS implementation	Campus model + basic algorithms
2	Complete UCS and A* implementation, test all algorithms	All search algorithms working
3	Build user interface, add information services, conduct algorithm comparison	Complete working system
4	Write report, prepare presentation, finalize documentation	Final submission

Submission Requirements

- **GitHub Repository**: Public repo with comprehensive documentation
- **Video Demo**: 5-minute screencast showing system capabilities
- **Technical Report**: PDF format, properly cited, following IEEE format
- **Source Code**: Well-commented, modular implementation
- **Test Cases**: Comprehensive test suite with validation results

Submission Deadline: [Insert specific date and time] **Late Penalty**: 10% per day, maximum 3 days extension