# Artificial Intelligence Project Report

## 🎯 Project Objective

In this project, the classical Artificial Intelligence problem known as the Tower of Hanoi is modeled as a state-space search problem. The goal is to reach the solution using the minimum number of steps, by implementing the Breadth-First Search (BFS) algorithm.

## 🧩 Problem Definition

The Tower of Hanoi involves moving disks of different sizes between three pegs, following specific rules:  
- Only one disk may be moved at a time.  
- A larger disk cannot be placed on top of a smaller disk.  
- The goal is to move all disks from the first peg to the third peg in order.

## ⚙️ AI Method: Breadth-First Search (BFS)

- BFS expands nodes level by level.  
- It guarantees the shortest path to the goal.  
- Ideal for problems where a minimal number of steps is required.

## 🧠 Data Structures

State:  
```  
typedef struct {  
 int pegs[3][MAX\_DISKS];  
 int heights[3];  
} State;  
```  
Node:  
```  
typedef struct Node {  
 State state;  
 struct Node \*parent;  
 int depth;  
} Node;  
```

## 📁 File Structure

| File | Description |  
|------|-------------|  
| main.c | Entry point of the program. |  
| data\_types.h | Contains core data structures. |  
| GRAPH\_SEARCH.c/h | Handles open list operations for graph search. |  
| HashTable.c/h | Manages visited states using a hash table. |  
| SpecificToProblem.c | Defines Hanoi-specific rules and state transitions. |  
| Standart\_Search.c | Implements the BFS algorithm. |

## 🔁 Problem-Specific Functions

- `initial\_state()` – Initializes the problem state.  
- `is\_goal()` – Checks if the goal has been reached.  
- `successors()` – Generates valid next states.  
- `print\_state()` – Prints a state to the console.

## 🔍 BFS Function

```  
void BFS(State start, int num\_disks);  
```  
- Initializes open list and visited hash table.  
- Begins with the start state.  
- Expands each node to find goal state.  
- Prints the path once goal is reached.

## ✅ Execution Steps

1. Initialize disks on the first peg.  
2. Use BFS to explore valid moves.  
3. Reach goal state in minimal steps.  
4. Print each intermediate step.

## 🧪 Sample Output (3 Disks)

```  
State:  
 Peg 0: 3 2 1  
 Peg 1:  
 Peg 2:  
  
State:  
 Peg 0: 3 2  
 Peg 1:  
 Peg 2: 1  
  
...  
  
State:  
 Peg 0:  
 Peg 1:  
 Peg 2: 3 2 1  
```

## 📌 Conclusion

In this project, the Tower of Hanoi problem is successfully solved using Breadth-First Search (BFS). Since BFS guarantees the shortest solution, it is well-suited for this type of state-space problem.

## 🔚 Extensions

- The algorithm can be enhanced with DFS, A\* or other strategies.  
- Disk number can be increased to observe performance.  
- Visual representation of the graph/tree can be added.