## Indian Institute of Information Technology Surat

****

# Lab Report on

# High Performance Computing (CS 602) Practical

**Submitted by**

### [RAHUL KUMAR SINGH] (UI21CS44)

**Course Faculty**

### Dr. Sachin D. Patil

## Department of Computer Science and Engineering

## Indian Institute of Information Technology Surat

## Gujarat-394190, India

**Jan-2024**

## Lab No: 5

**Aim: To Develop an efficient algorithm to solve the 10×10 chessboard problem, placing 10 Queens or Horses without conflicts, and analyze time complexity and execution time for both individual and all possible solutions.**

**Description:**

* Problem Statements:

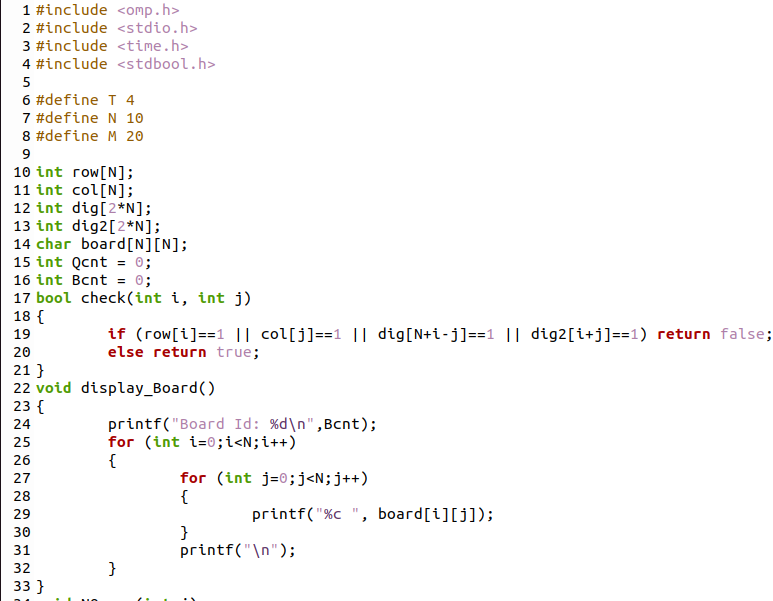
i) 10 Queens: Avoid striking each other.

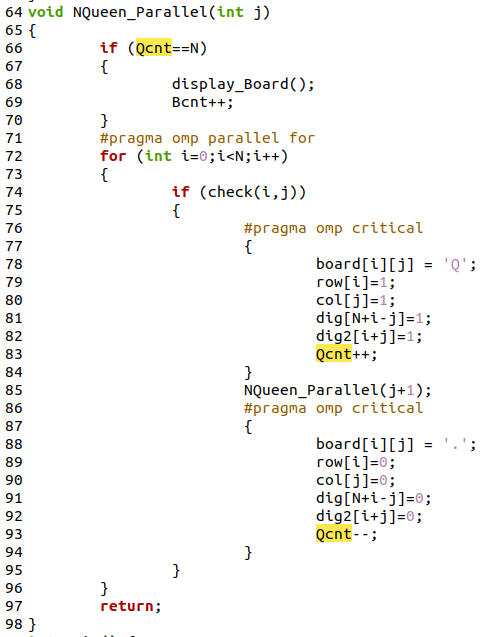
* Algorithm: Recursive placement with OpenMP parallelization.
* Analyze time complexity and execution time for individual and all solutions.

ii) 10 Horses: Identify maximum placements without conflicts.

* Algorithm: Parallelized approach using OpenMP for horse placement.
* Analyze time complexity and execution time for individual and all solutions.
* Queen Placement: Backtracking algorithm ensuring no conflicts.
* Horse Placement: Parallelized approach to maximize placements without conflicts.
* OpenMP: Utilized for parallel constructs, enhancing computational efficiency.
* Dynamic and Static Scheduling: Applied for load balancing in parallel sections.

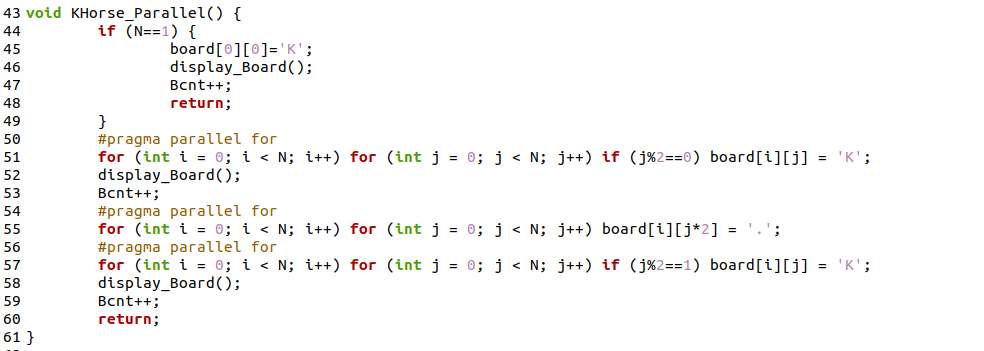
## Source Code:

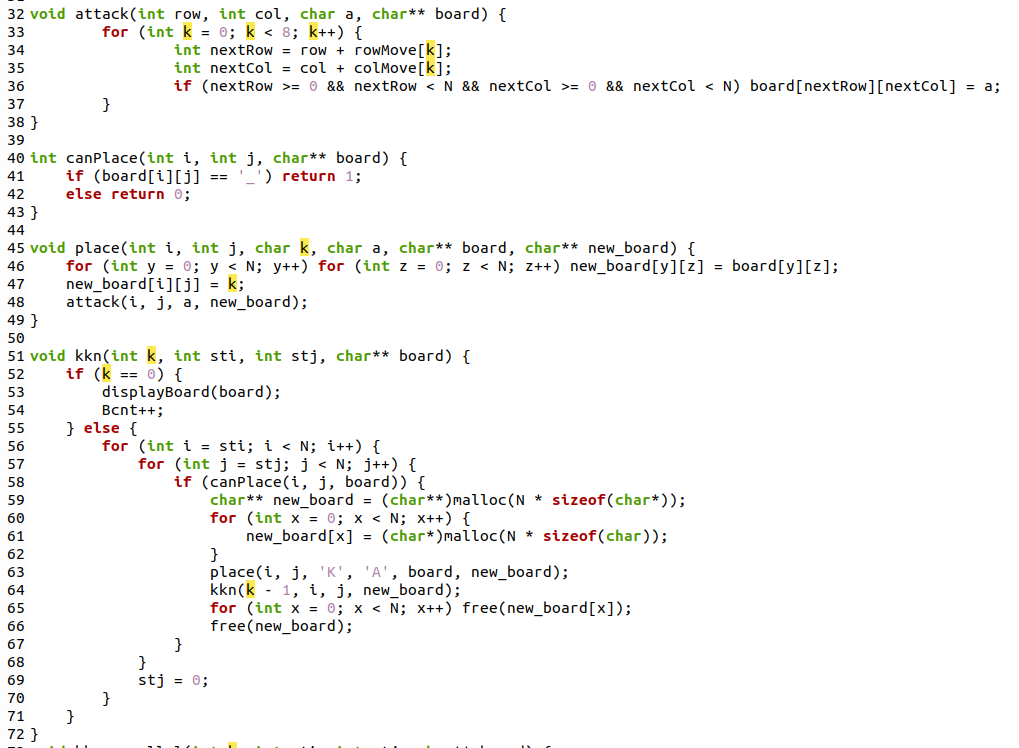
**Task 1:  
**

****

****

## Task 2:

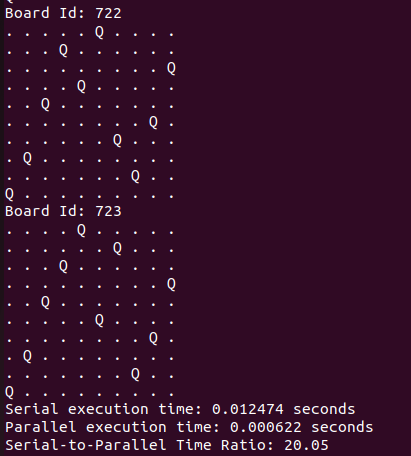




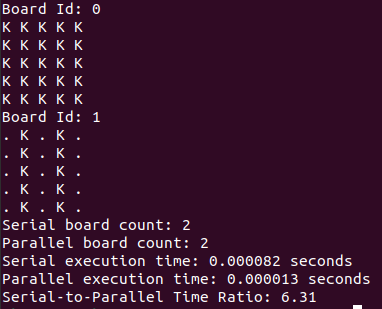
## 

## Output:

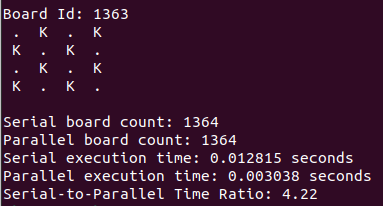
**Task 1:**

****

**Task 2:**

**For Maximum:  
**

**For All Possible Combination:**

****

## Conclusion:

* Utilized OpenMP for parallelizing Queen placement on a 10×10 chessboard.
* Explored static, dynamic, guided, auto, and runtime scheduling options for load balancing.
* Investigated the maximum number of non-conflicting Horse placements (2 for N>1 and 1 for N=1).
* Analyzed time complexity and execution time for solutions.
* Implemented dynamic memory allocation for efficient array usage.
* Ensured proper memory deallocation to prevent memory leaks.