**A PRELIMINARY REPORT ON**

**Design and Analysis of Algorithms Mini Project**

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE ACADEMIC OF

**FOURTH YEAR OF COMPUTER ENGINEERING**

**SUBMITTED BY**

**Harsh Bhattad BBCO20104**



**DEPARTMENT OF COMPUTER ENGINEERING**

**DR. D.Y.PATIL INSTITUTE OF ENGINEERING, MANAGEMENT & RESEARCH**

**AKURDI, PUNE 411044**

**SAVITRIBAI PHULE PUNE UNIVERSITY**

AY 2023 -2024



**CERTIFICATE**

This is to certify that the Mini Project report of

**Design and Analysis of Algorithms**

Submitted by

**Akshay Paradke BBCO20164**

**is a bonafide student of this institute and the work has been carried out by them under the supervision of Mrs. Deepali Jawaleand it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University, for the award of the Fourth year degree of Computer Engineering.**

**Mrs. Deepali Jawale** **M/s. P. P. Shevatekar**

Guide Head

Department of Computer Engineering Department of Computer Engineering

Place: Pune

Date:

**ACKNOWLEDGEMENT**

First and foremost, I would like to thank my guide for this Mini Project, **Mrs. Deepali Jawale** for the valuable guidance and advice. She inspired us greatly to work in this seminar. Her willingness to motivate us contributed tremendously to our seminar work. I also would like to thank her for showing me some examples that related to the topic of my Mini Project.

Apart from our efforts, the success of any seminar depends largely on the encouragement and guidelines of many others. So, we take this opportunity to express my gratitude to **M/s. P. P. Shevatekar**, Head of the Department of Computer Engineering, Dr. D Y Patil Institute of Engineering, Management And Research, Akurdi has been instrumental in the successful completion of this seminar work.

The guidance and support received from all the members who contributed and who are contributing to this seminar work were vital for the success of the seminar. I am grateful for their constant support and help.

**Harsh Bhattad**

Student Name

(B.E. COMPUTER ENGG.)

**TABLE OF CONTENT**

|  |  |  |
| --- | --- | --- |
| **Sr No.** | **Content** | **Page No.** |
| 1 | Introduction | 6 - 8 |
|  | 1.1. Introduction | 6 |
|  | 1.2. Problem Statement | 7 |
|  | 1.3. Objectives | 7 |
| 2 | Methodology | 9 - 10 |
| 3 | Implementation | 11 - 17 |
| 4 | Conclusion | 18 |

**CHAPTER 1: INTRODUCTION**

**1.1. INTRODUCTION**

In the realm of computer science and parallel computing, matrix multiplication stands as a fundamental operation with a wide range of applications across various domains, including mathematics, physics, engineering, and computer graphics. The efficiency of matrix multiplication can significantly impact the performance of numerous algorithms and computations. In this programming project, we will delve into the world of matrix multiplication, exploring two distinct approaches to this problem: a traditional single-threaded implementation and a more advanced multithreaded approach.

The primary objective of this project is to design and develop a program capable of multiplying matrices, providing not only a correct solution but also a means of comparing the performance of two different paradigms: one thread per row and one thread per cell. By implementing these approaches, we aim to demonstrate the advantages and trade-offs associated with multithreaded matrix multiplication and analyze how they impact computation time, resource utilization, and overall system efficiency.

Throughout this project, we will investigate the design, implementation, and performance analysis of both single-threaded and multithreaded matrix multiplication methods, shedding light on the practical considerations and insights that arise from this comparative study. Ultimately, this exploration will enable us to gain a deeper understanding of how parallel computing can enhance the efficiency of mathematical operations and contribute to improved performance in a wide range of applications.

**1.2. PROBLEM STATEMENT**

The problem statement for this project is to developing a program to perform matrix multiplication and to explore the advantages and trade-offs of multithreaded matrix multiplication using two distinct approaches: one thread per row and one thread per cell. Your program should provide a comprehensive analysis of the performance of these methods to evaluate their efficiency in solving this fundamental computational problem.

**1.3. OBJECTIVE**

The objective of this project is to develop a comprehensive program that accomplishes the following:

Matrix Multiplication: Implement a functional matrix multiplication algorithm that accurately multiplies two matrices, allowing the user to input matrix dimensions and elements.

Multithreaded Matrix Multiplication: Explore the benefits and trade-offs of multithreaded matrix multiplication by implementing two distinct threading strategies:

"One Thread per Row": Create and employ threads to calculate matrix rows concurrently.

"One Thread per Cell": Leverage threading to compute individual matrix cells in parallel.

Performance Analysis: Conduct a rigorous performance analysis to assess the efficiency and effectiveness of both single-threaded and multithreaded matrix multiplication methods. This analysis should encompass key metrics, such as execution time, CPU and memory usage, scalability with varying matrix sizes, and considerations for potential bottlenecks and resource utilization.

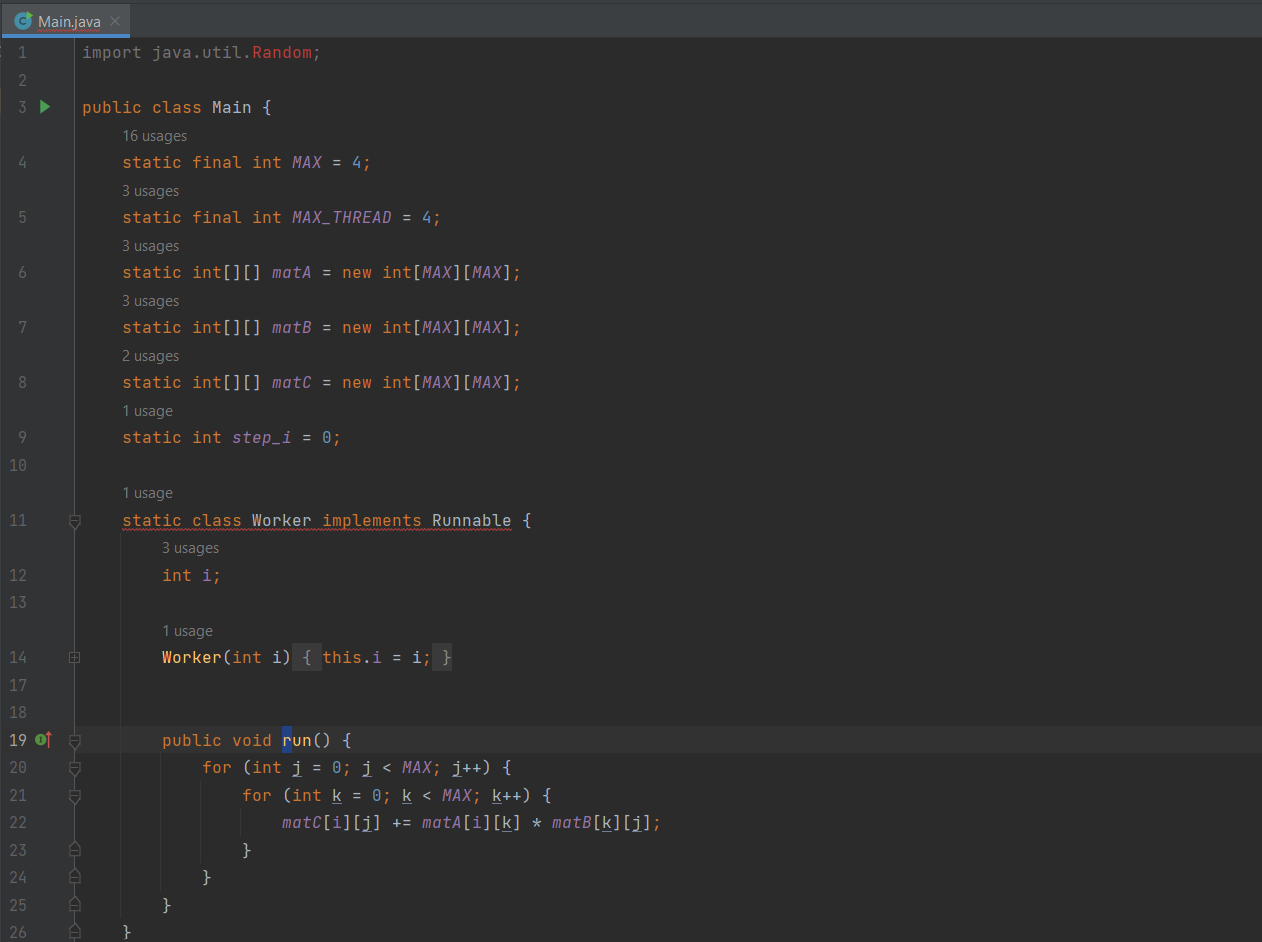
Comparison: Compare and contrast the performance of the implemented methods, providing insights into the advantages and limitations of multithreading for matrix multiplication.

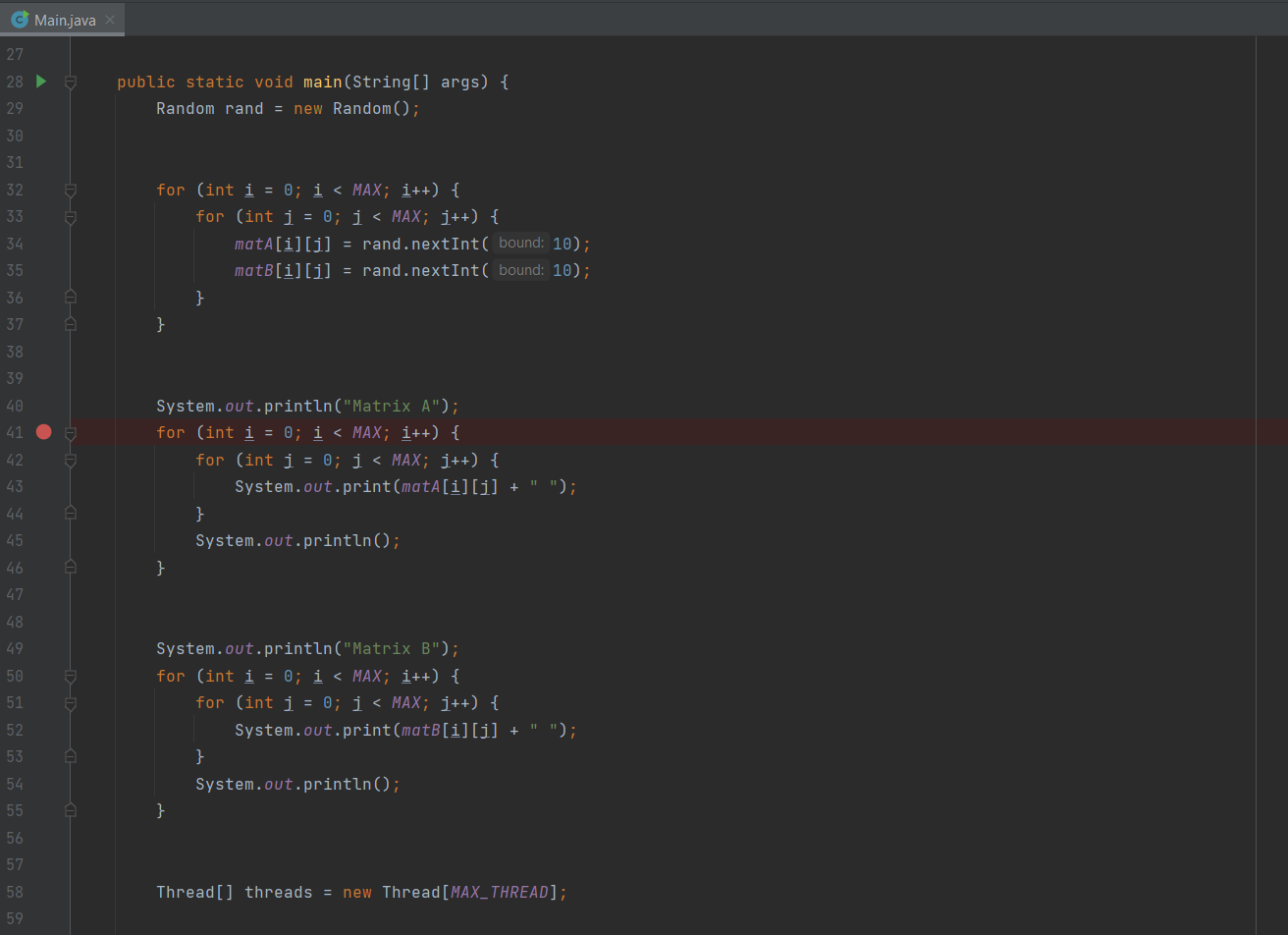
Through this project, we aim to achieve a deeper understanding of the computational implications of multithreading in the context of matrix multiplication, and to provide a clear and well-documented comparison of these approaches, enabling informed decisions on when and how to employ multithreading for this fundamental mathematical operation.

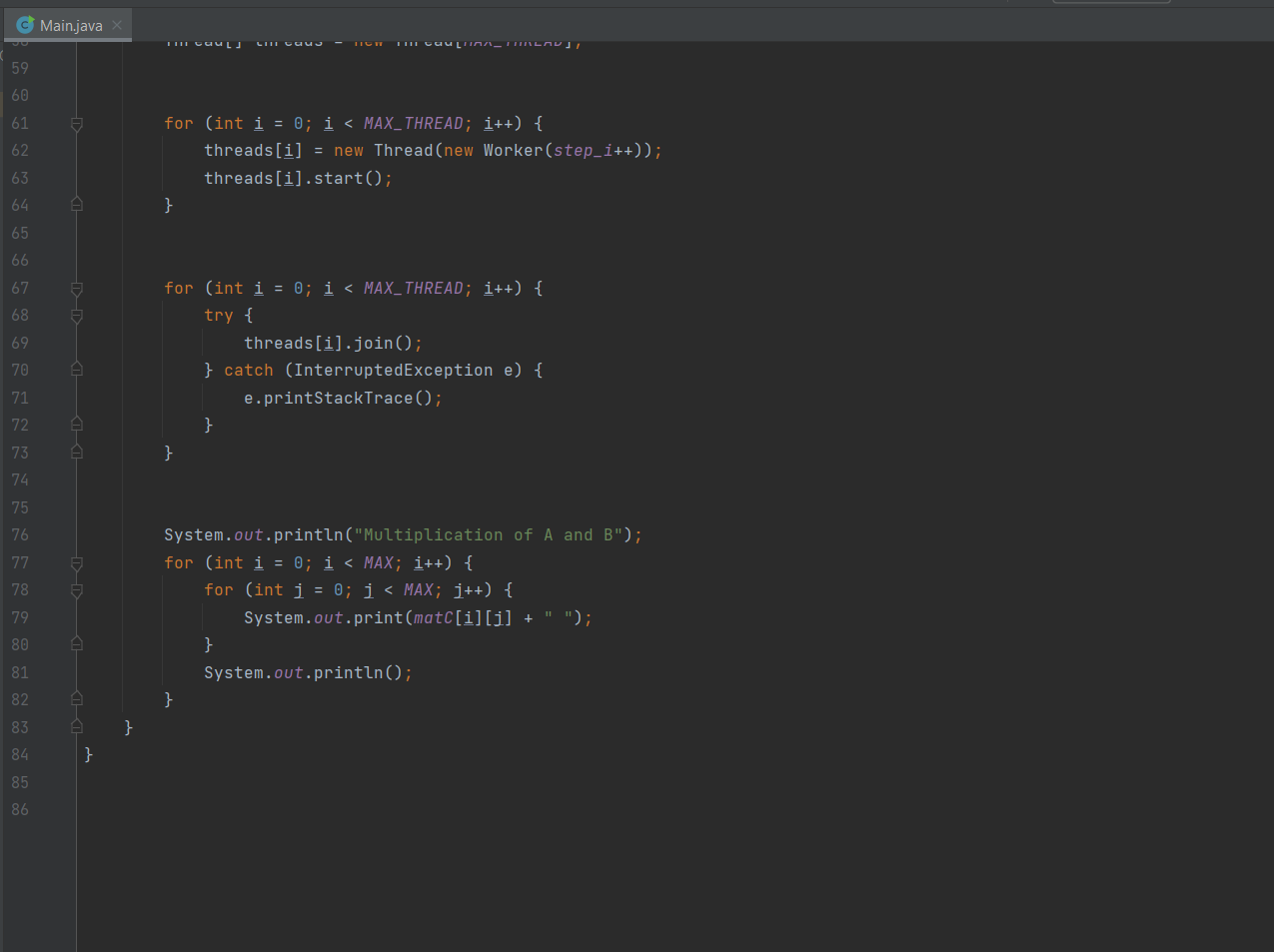
**CHAPTER 2: METHODOLOGY**

1. **Problem Understanding:** Begin by thoroughly understanding the problem statement and the specific requirements for implementing matrix multiplication and analyzing different multithreaded approaches. Identify the input parameters, expected output, and performance metrics to be measured.
2. **Algorithm Design:**
   1. Design a matrix multiplication algorithm that adheres to the principles of linear algebra.
   2. For single-threaded multiplication, follow the conventional algorithm.
   3. For multithreaded multiplication, design separate algorithms for one thread per row and one thread per cell approaches. Ensure that the parallelization is correctly managed to prevent data race conditions.
3. **Software Environment Setup:**
   1. Choose a programming language and development environment suitable for the task, considering support for multithreading.
   2. Import any necessary libraries or modules for matrix operations and threading.
4. **User Interface Development:**
   1. Create a user-friendly interface to input matrix dimensions and elements.
   2. Implement functionality for the user to choose between single-threaded and multithreaded approaches.
5. **Matrix Generation:**
   1. Develop a method for generating random matrices or allow user-defined input.
   2. Ensure that the generated matrices adhere to the dimensions specified by the user.
6. **Single-Threaded Implementation:**
   1. Write the code for the single-threaded matrix multiplication, ensuring correctness.
   2. Measure the execution time of the single-threaded approach using appropriate timers.
7. **Multithreaded Implementation:**
   1. For the one thread per row approach, create and manage threads such that each thread computes a row of the resulting matrix.
   2. For the one thread per cell approach, create threads that calculate individual cells in the result matrix.
   3. Implement thread synchronization to prevent data race conditions or inconsistencies.
   4. Measure the execution time for both multithreaded methods.
8. **Performance Analysis:**
   1. Execute the matrix multiplication methods with different input sizes, ranging from small to large matrices.
   2. Collect performance metrics for each method, including execution time, CPU usage, and memory consumption.
   3. Analyze the scalability of multithreaded approaches and identify any bottlenecks or resource limitations.
9. **Comparison and Evaluation:**
   1. Compare the performance of single-threaded and multithreaded approaches based on the collected metrics.
   2. Assess the trade-offs in terms of speedup, efficiency, and resource utilization.
   3. Provide insights into when each approach is most beneficial.
10. **Documentation and Reporting:**
    1. Document the design, implementation details, and performance analysis methodology.
    2. Create a clear and concise report describing the experiment, the results, and your conclusions.
    3. Include charts or graphs to visually represent the performance comparisons.
11. **Optimization (Optional):**
    1. Explore potential optimization strategies for improving the performance of multithreaded approaches.
    2. Experiment with different thread management techniques, data structures, or algorithm modifications.
12. **Testing and Validation:**
    1. Validate the correctness of your program by comparing its output with a reliable matrix multiplication library or a known solution.
    2. Ensure that the program behaves as expected and handles user input appropriately.
13. **Presentation (Optional):**
    1. If applicable, prepare a presentation or demonstration to communicate your findings and insights to an audience.
14. **Final Report and Conclusion:**
    1. Compile all the results, findings, and insights into a final report.
    2. Conclude the report by summarizing the advantages and limitations of each approach and the implications for practical use.
15. **Project Completion:**
    1. Organize and submit all project deliverables, including code, documentation, and the final report.

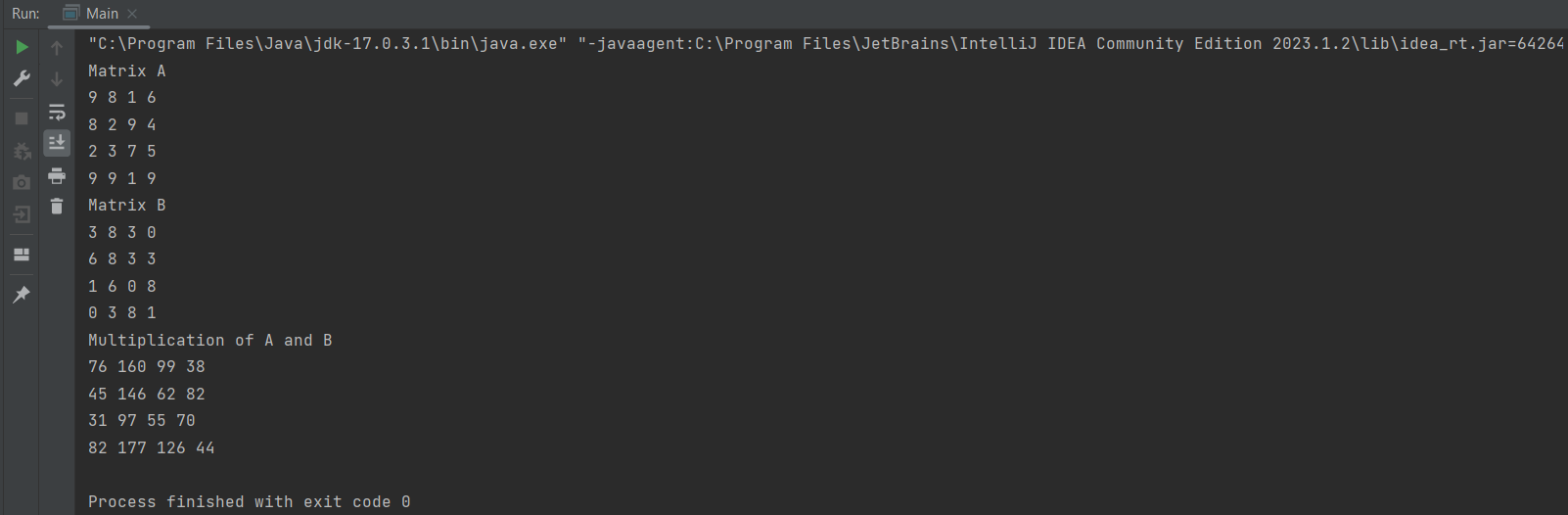
**CHAPTER 3: IMPLEMENTATION**

****

****

****

**Solution**

****

**CHAPTER 4: CONCLUSION**

In the pursuit of optimizing matrix multiplication, this project has taken us on a journey through the realms of both single-threaded and multithreaded computing, shedding light on the significance of choosing the right approach for this fundamental mathematical operation. Through the implementation of matrix multiplication and the exploration of two distinct multithreaded strategies—one thread per row and one thread per cell—we have gained valuable insights into their respective strengths and weaknesses.