COMP3432 – OPERATING SYSTEMS PROJECT SPRING 2025

IŞIK UNIVERSITY
COMPUTER SCIENCE AND ENGINEERING

OBJECTIVE

The objective of this project is to design and implement a memory-efficient, multithreaded Java program that performs large-scale matrix multiplication using file-based data access. The project aims to help students apply key operating system concepts—such as concurrency, thread partitioning, synchronization, and performance evaluation—by simulating a real-world computational task.

PROJECT DETAILS

Imagine a scientist working on climate simulations who needs to perform massive matrix multiplications to process environmental data collected from thousands of sensors. Each simulation run involves multiplying large matrices, and doing it sequentially takes far too long. To solve this challenge, the scientist enlists the expertise of you (a team of two students).

Your mission is to build a multithreaded Java program that accelerates these computations by leveraging the power of parallel processing. *Instead of loading the entire matrices into memory, your solution will read only the relevant parts of the data from a file. Each thread will compute a specific portion of the result matrix. Matrix B will be loaded once into memory and shared among threads, while each thread independently reads its assigned rows of Matrix A*. This project will challenge your understanding of operating systems concepts such as concurrency, thread management, synchronization, and memory-efficient computation. Your team will also compare the performance of your multithreaded approach against a traditional single-threaded version which you need to implement as well. You will also document your results and division of responsibilities by using the report template shared as a companion to this project file.

To test your implementation, **a script is shared**. Use the one suited for your operating system (*Windows (.bat) and Linux/MacOS (.sh)*). You must run the script to run the implemented code to generate a benchmark report file. The results of all the runs will be saved in this auto generated file which you will use to analyze the behavior of your program. Discuss your analysis in the report.

A code skeleton is shared with you. **The codes that are already there must not be changed**. If you change the already given code, your script file may fail to run. Also, you need to put the .java files for single and multithreaded solutions along with the sample input file and the script file into the same folder. Then go to your CLI and run the script file.

Follow the instructions given below.

Directory Structure

The directory structure must be the same as below.

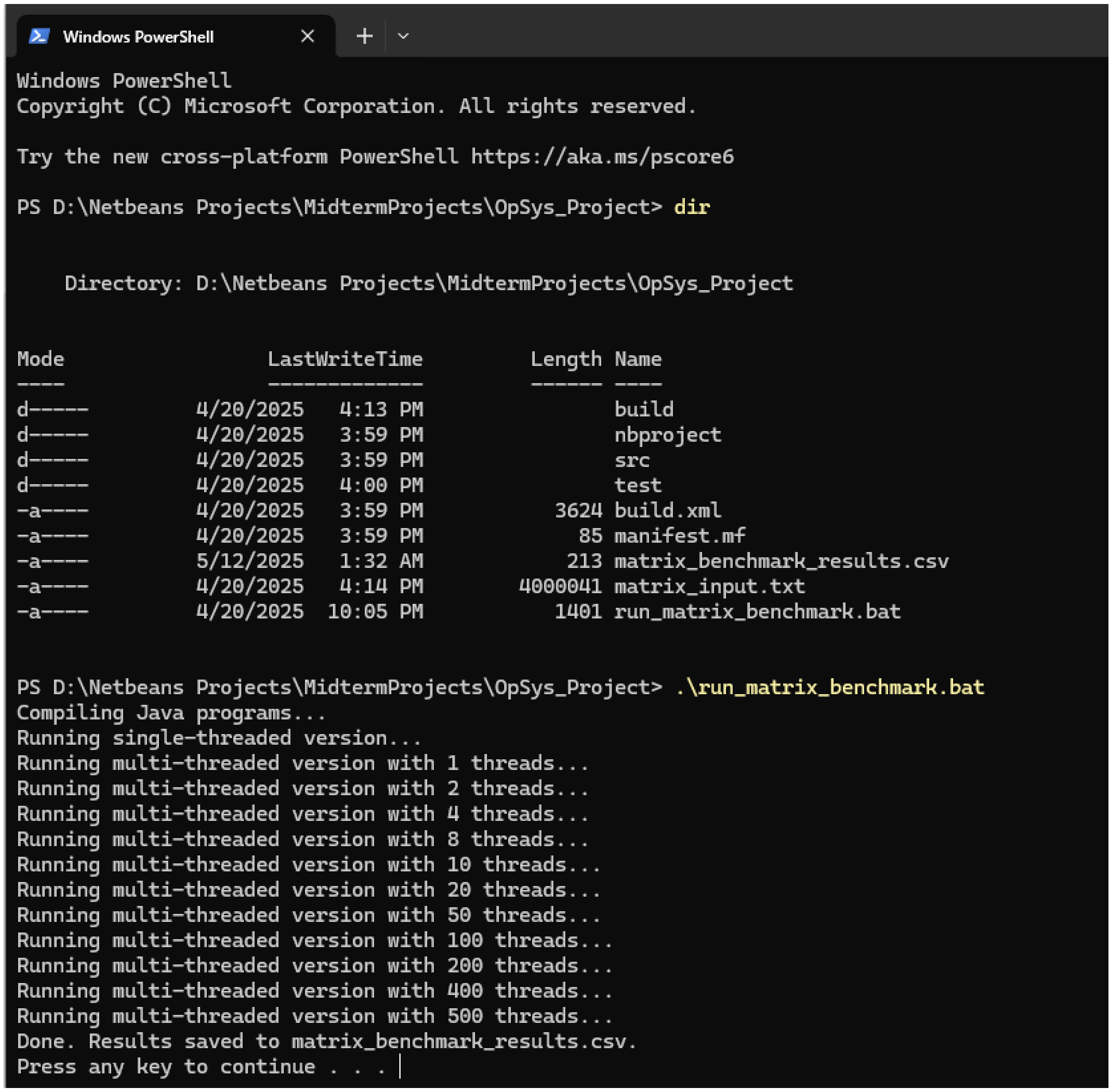
```
/YourProjectFolder
│
├── run_matrix_benchmark.bat
├── matrix_input.txt
├── src/
│   └── opsys_project/
│           ├── MatrixMultiplier_SingleThread.java
│           └── MatrixMultiplier_MultiThread.java
│
```

1. **DO NOT** change the method signatures or printed output format in the skeleton code.
   a. The batch script depends on the console output of your program.
2. **DO NOT** rename classes or packages
   a. They are required to be in the “opsys_project” package
3. Use the following command in **CLI** of you Operating System to run the benchmark
   a. In Windows CMD to run the benchmark: **run_matrix_benchmark.bat**
   b. In Windows PowerShell: **.\run_matrix_benchmark.bat**
   c. In UNIX-like systems Terminal:
      i. **chmod +x run_matrix_benchmark.sh**
      ii. **./run_matrix_benchmark_mac.sh**

   The script will:

      i. Compile your Java files
      ii. Run your programs with different thread counts
      iii. Log execution times into **matrix_benchmark_results.csv**
4. Do not modify **run_matrix_benchmark.bat** or the **run_matrix_benchmark.sh** file

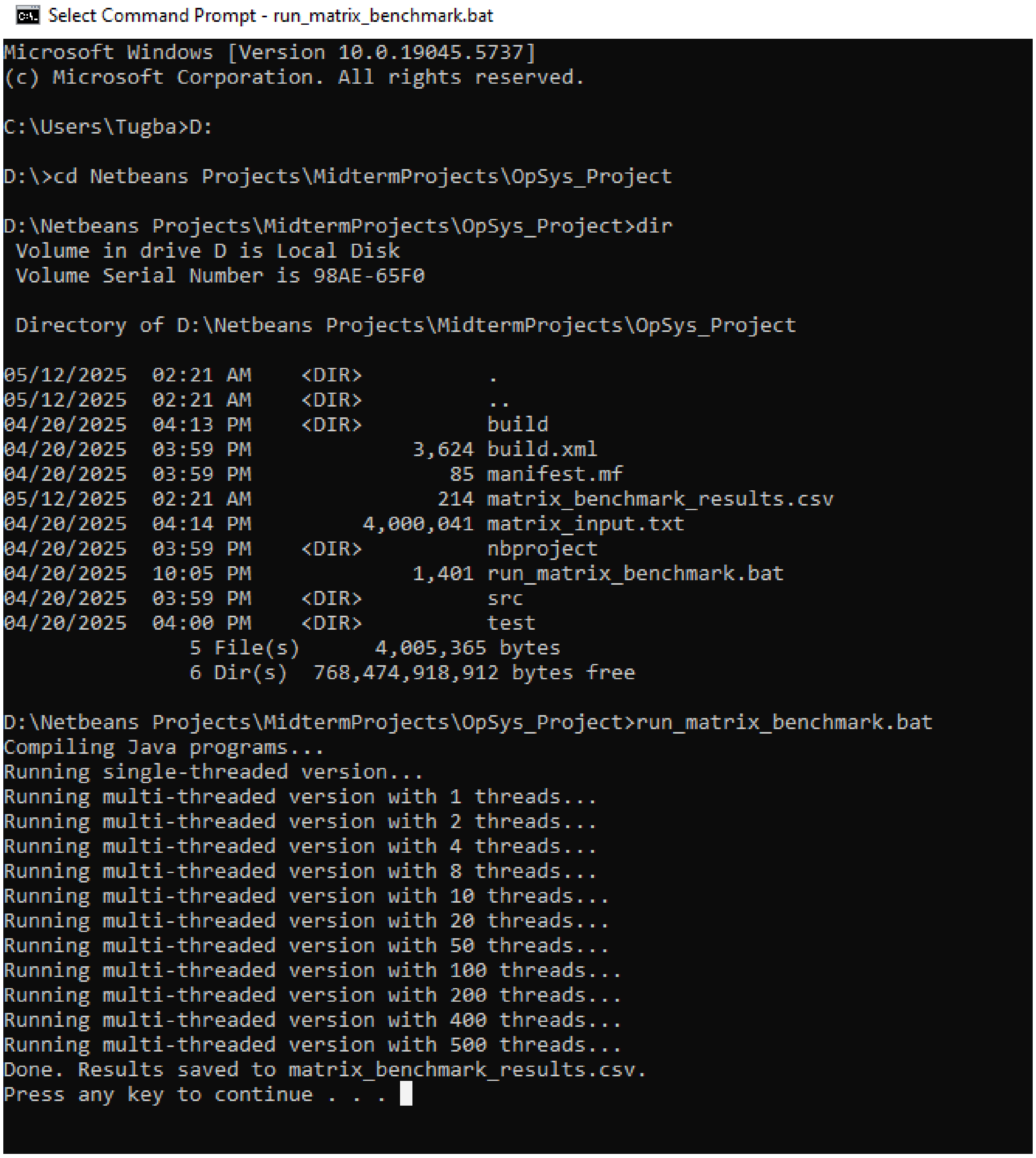
Example run from Windows PowerShell:



Windows PowerShell

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS D:\Netbeans Projects\MidtermProjects\OpSys_Project> dir


    Directory: D:\Netbeans Projects\MidtermProjects\OpSys_Project


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         4/20/2025   4:13 PM                build
d-----         4/20/2025   3:59 PM                nbproject
d-----         4/20/2025   3:59 PM                src
d-----         4/20/2025   4:00 PM                test
-a----         4/20/2025   3:59 PM           3624 build.xml
-a----         4/20/2025   3:59 PM             85 manifest.mf
-a----         5/12/2025   1:32 AM            213 matrix_benchmark_results.csv
-a----         4/20/2025   4:14 PM        4000041 matrix_input.txt
-a----         4/20/2025  10:05 PM           1401 run_matrix_benchmark.bat


PS D:\Netbeans Projects\MidtermProjects\OpSys_Project> .\run_matrix_benchmark.bat
Compiling Java programs...
Running single-threaded version...
Running multi-threaded version with 1 threads...
Running multi-threaded version with 2 threads...
Running multi-threaded version with 4 threads...
Running multi-threaded version with 8 threads...
Running multi-threaded version with 10 threads...
Running multi-threaded version with 20 threads...
Running multi-threaded version with 50 threads...
Running multi-threaded version with 100 threads...
Running multi-threaded version with 200 threads...
Running multi-threaded version with 400 threads...
Running multi-threaded version with 500 threads...
Done. Results saved to matrix_benchmark_results.csv.
Press any key to continue . . .
```

Example run from Windows CMD:



Select Command Prompt - run_matrix_benchmark.bat

```
Microsoft Windows [Version 10.0.19045.5737]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Tugba>D:

D:\>cd Netbeans Projects\MidtermProjects\OpSys_Project

D:\Netbeans Projects\MidtermProjects\OpSys_Project>dir
 Volume in drive D is Local Disk
 Volume Serial Number is 98AE-65F0

 Directory of D:\Netbeans Projects\MidtermProjects\OpSys_Project

05/12/2025  02:21 AM    <DIR>          .
05/12/2025  02:21 AM    <DIR>          ..
04/20/2025  04:13 PM    <DIR>          build
04/20/2025  03:59 PM             3,624 build.xml
04/20/2025  03:59 PM                85 manifest.mf
05/12/2025  02:21 AM               214 matrix_benchmark_results.csv
04/20/2025  04:14 PM         4,000,041 matrix_input.txt
04/20/2025  03:59 PM    <DIR>          nbproject
04/20/2025  10:05 PM             1,401 run_matrix_benchmark.bat
04/20/2025  03:59 PM    <DIR>          src
04/20/2025  04:00 PM    <DIR>          test
               5 File(s)      4,005,365 bytes
               6 Dir(s)  768,474,918,912 bytes free

D:\Netbeans Projects\MidtermProjects\OpSys_Project>run_matrix_benchmark.bat
Compiling Java programs...
Running single-threaded version...
Running multi-threaded version with 1 threads...
Running multi-threaded version with 2 threads...
Running multi-threaded version with 4 threads...
Running multi-threaded version with 8 threads...
Running multi-threaded version with 10 threads...
Running multi-threaded version with 20 threads...
Running multi-threaded version with 50 threads...
Running multi-threaded version with 100 threads...
Running multi-threaded version with 200 threads...
Running multi-threaded version with 400 threads...
Running multi-threaded version with 500 threads...
Done. Results saved to matrix_benchmark_results.csv.
Press any key to continue . . .
```

CONSTRAINTS

- **Form a two-student group. No individual work.**
- You **must only use Java's Thread class or implement the Runnable interface**. **DO NOT** use Callable, Future, or thread pools.

IŞIK UNIVERSITY
COMPUTER SCIENCE AND ENGINEERING

LEARNING OUTCOMES

- Develop Java applications using multiple threads.
- Read large datasets efficiently from a file, accessing only the required portions relevant to each thread's task.
- Design a solution that minimizes memory usage by loading only necessary data via threads.
- Measure and compare the performance of single-threaded versus multithreaded matrix multiplication to understand the benefits of concurrency.
- Experience how operating systems concepts like concurrency, I/O efficiency, and resource sharing apply to computational problems in domains such as science and engineering.

DELIVERABLES

Each student group is expected to submit the following in a single archive file (only **.zip** or **.rar** is allowed):

1. **Java Source Code**
   a. Includes both multithreaded and single-threaded versions.
   b. Clearly separates logic for reading matrix data from file and performing computation
2. **Log file**
   a. matrix_benchmark_results.csv
3. **Report file**