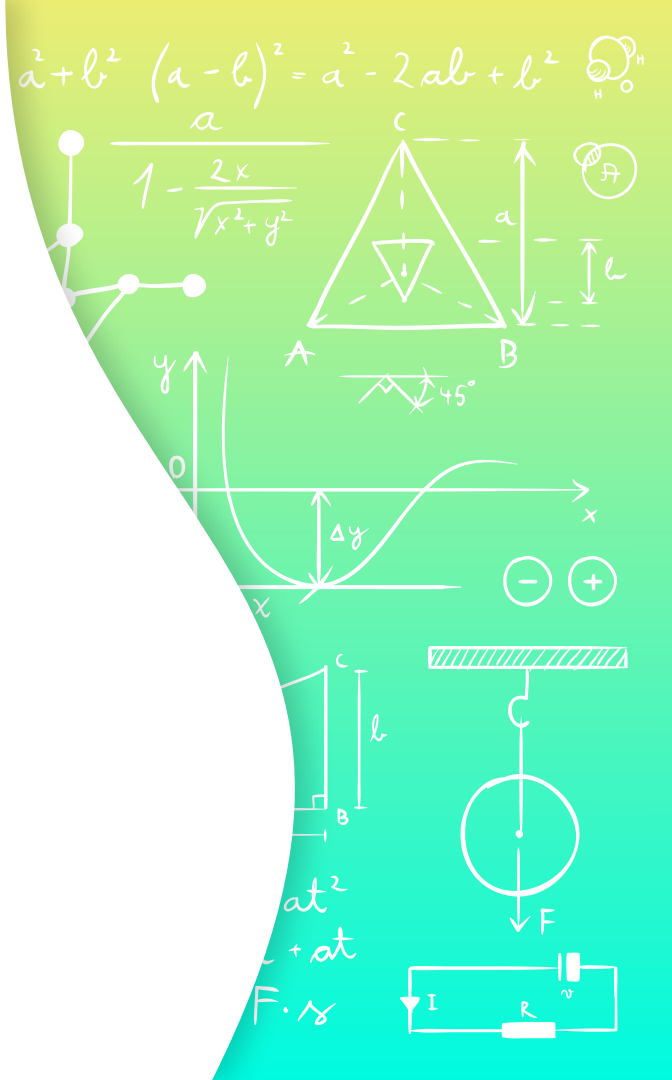
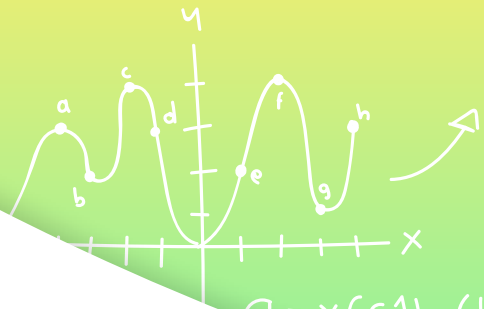
October 2024

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

CONCURRENT PROGRAMMING ASSIGNMENT SERIES 1

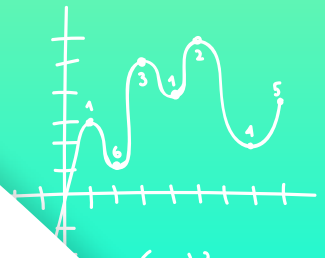
TEAM 11





$$e = f^2(x+4gh)^2(s).(x)^3 \div (gh)^2 - x^2$$
$$f = gh^2 + (s)(x+2h)^3 \times 4x^2(he)^3 + x^2 - 2x^2$$
$$g = x^2 \div (x)(2x)^2 + (hfe)^2 4x^3(3h)(f)^2(e)^2 + x^2 4s^2$$
$$h = efg^2 - (x)^2 + (3)^2(f)^3 + x(4x)^2$$
$$dh(x) \approx bc$$
$$(x)^2 \approx ab$$

$$a = x(s^1) + (h).(c) + (d)(ef)^2 = x^2$$
$$(h)(d) \div (s^1)(h^1)(b)^2 = \frac{4x^2 hd}{2s+4x}$$
$$ab = \frac{4x^2 + (ef)^2}{hc.s^2(x)_3}$$
$$dc = \frac{3x^2 + ab(s)^3}{xy^3 - (x)(s)^1}$$



$$(x)^2 \approx ab$$
$$(x) \approx bc$$

1.1

FIFO PIPES

An Overview of Pipe Implementation Using Threads in C.

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

MAIN IDEA

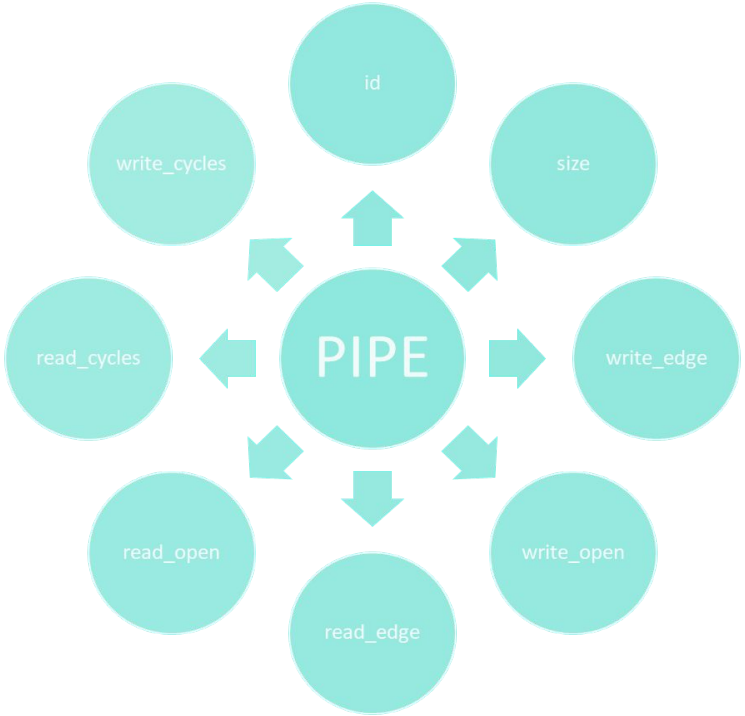
Implement a FIFO pipe mechanism using threads.

The FIFO Pipes ensure that data in the pipe is read in the same order it was written

Pipes are used to make threads communicate and exchange data

This makes synchronizing easier and our programs faster and more efficient

BUT WHAT IS A PIPE?



ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΙΑΣ

THE FUNCTIONS

```
int pipe_open(int size);
```

Creates a pipe

```
int pipe_write(int p, char c);
```

Writes one byte to pipe (p)

```
int pipe_writeDone(int p);
```

Closes pipe (p) for writing

```
int pipe_read(int p, char *c);
```

Reads one byte from pipe (p)

```
int my_read(int fd, void *buffer,
int size, int *left_read);
```

Reads from file

```
int my_write(int *fd, void
*buffer, int size);
```

Writes to file

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

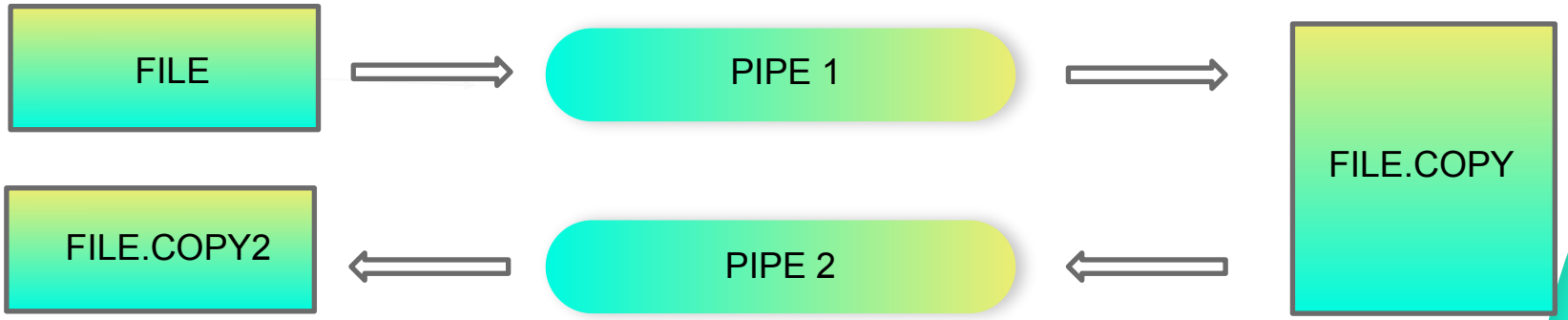
WHAT ARE THREADS RESPONSIBLE FOR?

THREAD 1

- Reads data from a file and writes it to pipe 1
- Reads from pipe 2 and writes to a new file

THREAD 2

- Reads from pipe 1 and writes to another file
- Reads from data from the created file and writes them

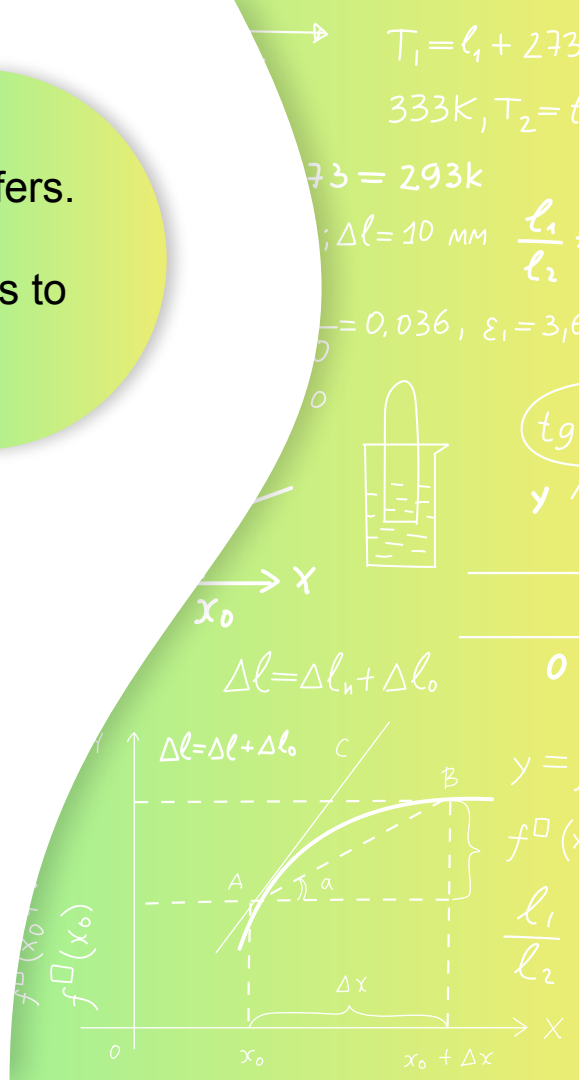


THE CHALLENGES

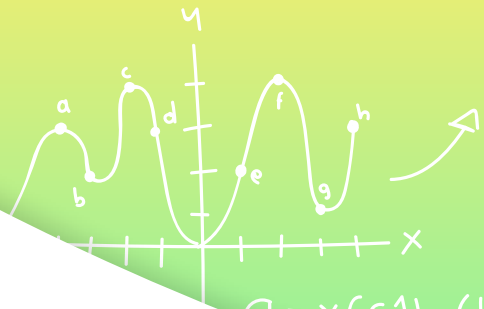
- Handling read/write operations in ring buffers.
- Synchronizing access to shared data between threads.

THE SOLUTIONS

- Properly managing pipe edges and cycles.
- Implementing conditions to handle concurrency.



ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

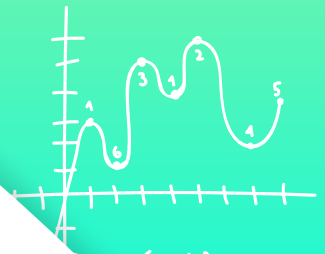


$$e = f^2(x+4gh)^2(s).(x)^3 \div (gh)^2 - x^2$$
$$f = gh^2 + (s)(x+2h)^3 \times 4x^2(he)^3 + x^2 - 2x^2$$
$$g = x^2 \div (x)(2x)^2 + (hfe)^2 4x^3 (3h)(f)^2(e)^2 + x^2 4s^2$$
$$h = efg^2 - (x)^2 + (3)^2(f)^3 + x(4x)^2$$
$$dh(x) \approx bc$$
$$(x)^2 \approx ab$$

1.2

PRIMALITY TESTER

$$a = x(s^1) + (h).(c) + (d)(ef)^2 = x^2$$
$$(h)(d) \div (s^1)(h^1)(b)^2 = \frac{4x^2 hd}{2s+4x}$$
$$ab = \frac{4x^2 + (ef)^2}{hc.s^2(x)_3}$$
$$dc = \frac{3x^2 + ab(s)^3}{xy^3 - (x)(s)^1}$$



$$(x)^2 \approx ab$$
$$(x) \approx bc$$

Implementation of a multi-thread primality tester

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

MAIN IDEA

This program takes positive integers as input and checks if they are prime or not.

Main is the coordinator of all threads.
Main assigns jobs to threads the so-called workers.

Workers busy wait for a job to be assigned to them.
When all jobs are finished, workers terminate.

IS_PRIME

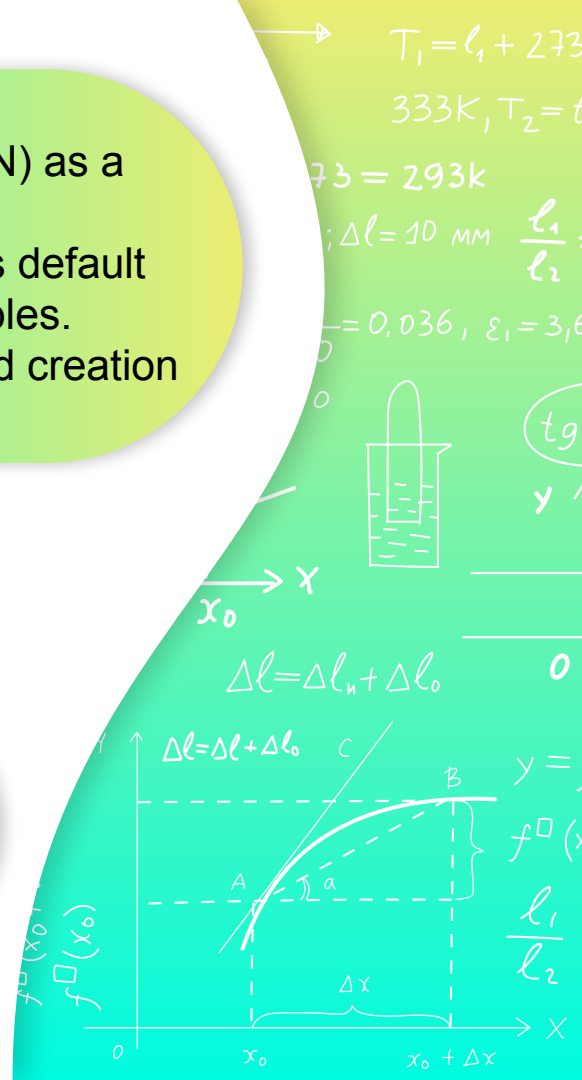
- Calculates if a number is a prime or not.

MAIN

- Takes the number of threads (N) as a command-line argument.
- Initializes each thread and sets default values for thread control variables.
- Error handling in case of thread creation failure.

WORKER

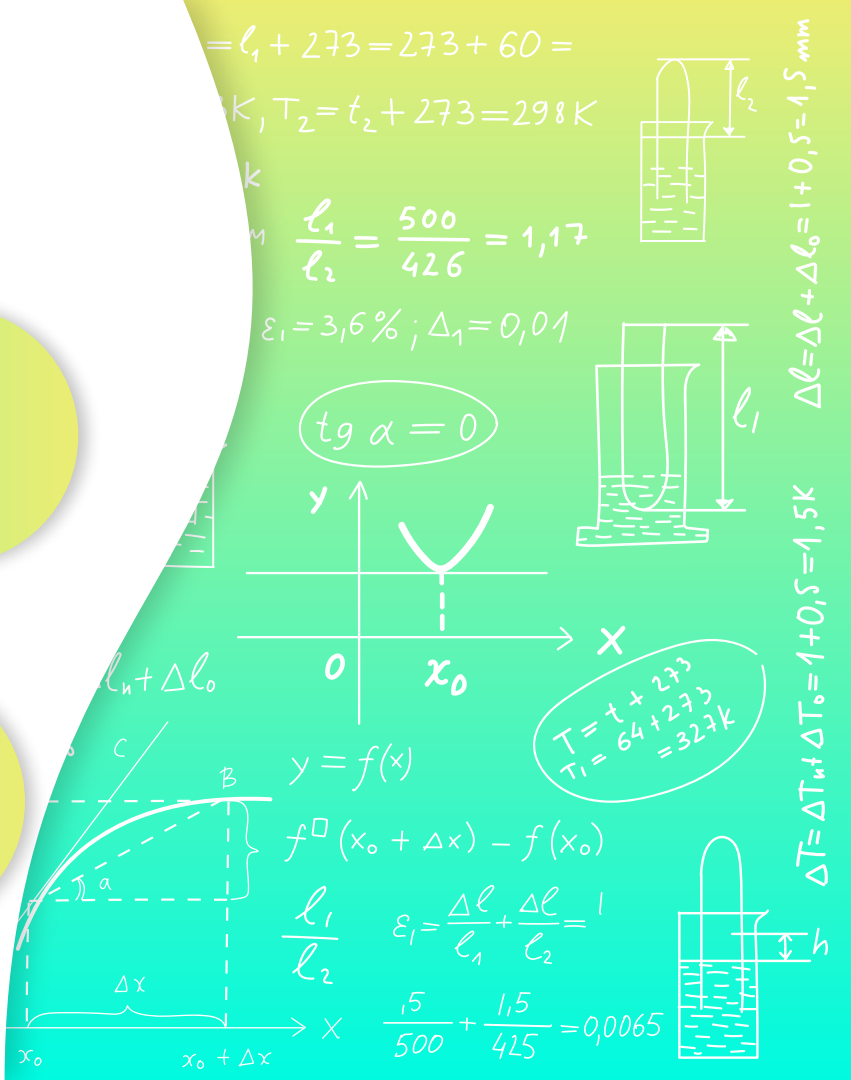
- Checks if the assigned number is prime.
- Updates thread status when work is completed.
- Terminates when requested by the main program.



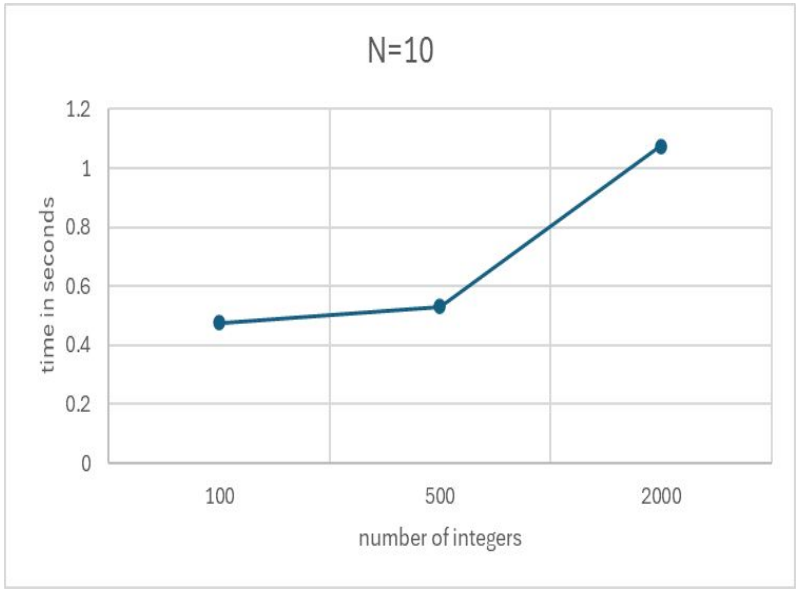
CHALLENGES

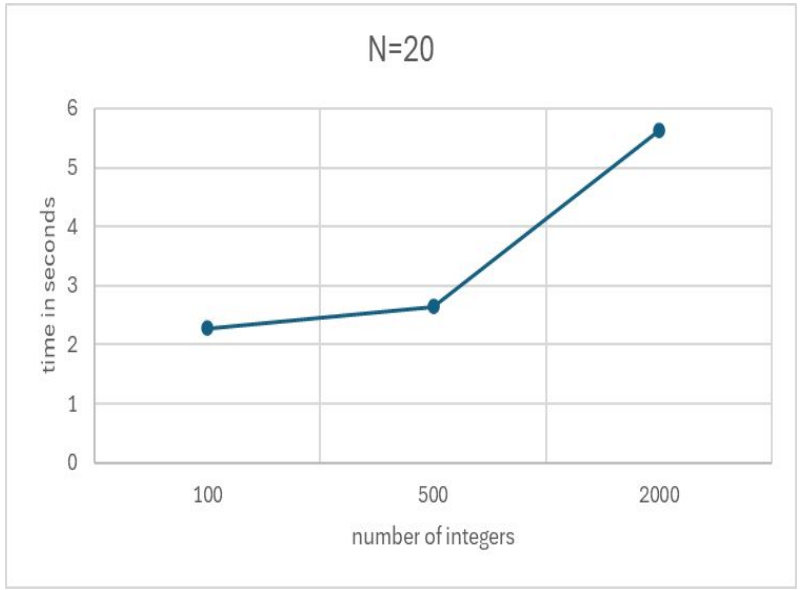
Synchronization Issues:
Avoiding race conditions when accessing shared resources.

Termination Handling:
Ensuring threads stop correctly without hanging or leaking resources.



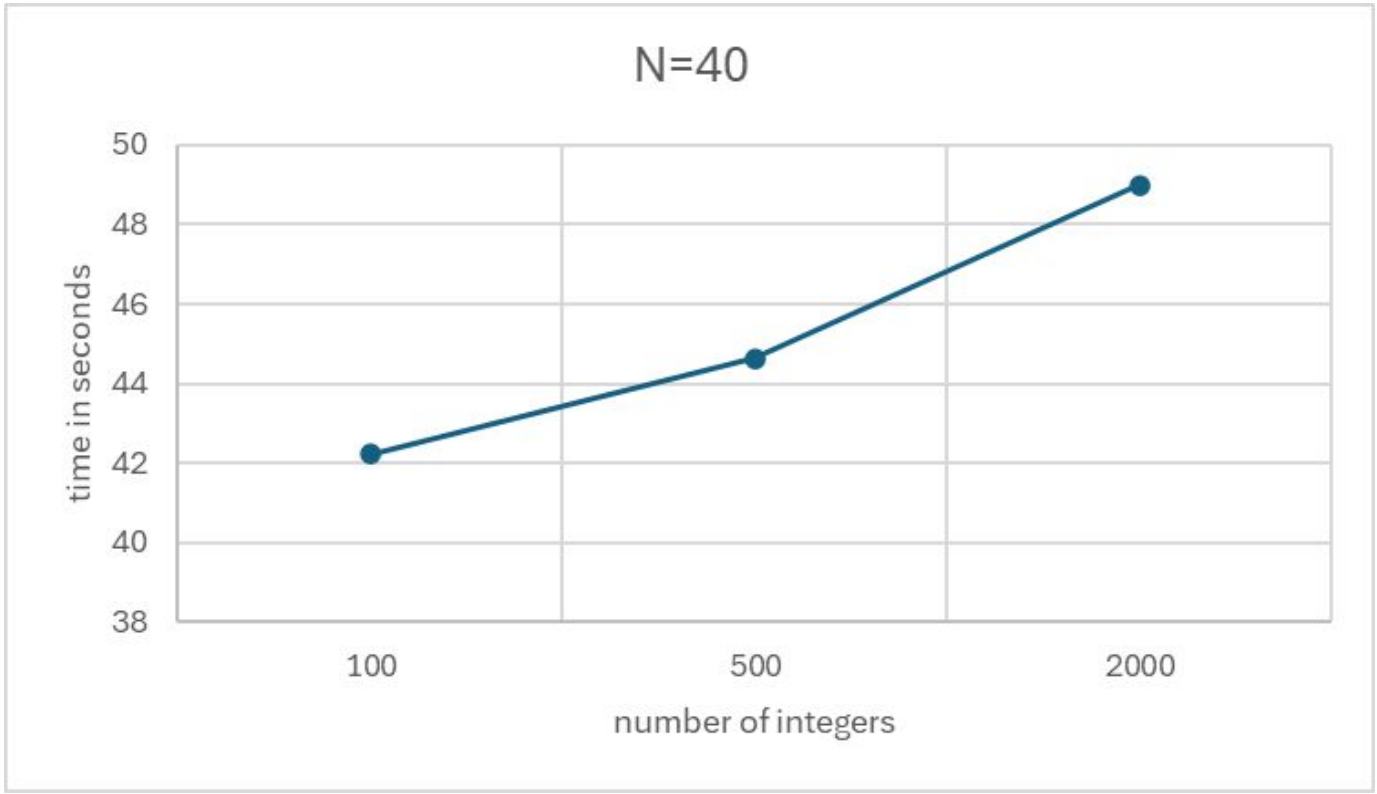
PERFORMANCE ANALYSIS



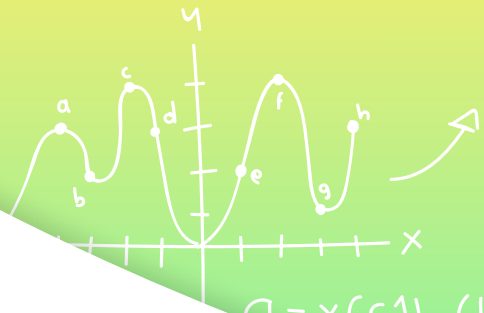


ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΙΑΣ

PERFORMANCE ANALYSIS

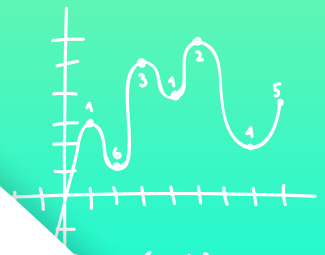


ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ



$$e = f^2(x+4gh)^2(s).(x)^3 \div (gh)^2 - x^2$$
$$f = gh^2 + (s)(x+2h)^3 \times 4x^2(he)^3 + x^2 - 2x^2$$
$$g = x^2 \div (x)(2x)^2 + (hfe)^2 4x^3(3h)(f)^2(e)^2 + x^2 4s^2$$
$$h = efg^2 - (x)^2 + (3)^2(f)^3 + x(4x)^2$$
$$dh(x) = bc$$
$$(x)^2 = ab$$

$$a = x(s^1) + (h).(c) + (d)(ef)^2 = x^2$$
$$(h)(d) \div (s^1)(h^1)(b)^2 = \frac{4x^2 hd}{2s+4x}$$
$$ab = \frac{4x^2 + (ef)^2}{hc.s^2(x)_3}$$
$$dc = \frac{3x^2 + ab(s)^3}{xy^3 - (x)(s)^1}$$



$$(x)^2 = ab$$
$$(x) = bc$$

1.3

EXTERNAL MERGESORT

A Parallel Approach to Sorting Large Files

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

MAIN IDEA

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

Efficiently sort large files using a parallel merge sort.

Implementation of External Merge Sort algorithm using threads to synchronize the whole process of merging a file.

Merge sort algorithm for external file sorting.

This makes synchronizing easier and our programs faster and more efficient.

THE FUNCTIONS

PARALLEL MERGESORT

- Parallel Execution: Uses threads to sort and merge file segments in parallel.
- Threshold: Splits files larger than MIN_SIZE (64) for parallel processing.
- Threading Structure:
  - Two threads per file split.
  - Splits until file segments are small enough.

MERGE SORT

Purpose: Sorts an array by recursively dividing and merging.

Key Steps:
1. Recursively split the array.
2. Use merge function to combine sorted halves.
3. Continue until array is sorted.

MERGE

Purpose: Merges two sorted arrays into one.
Process:
- Splits array into left and right halves.
- Compares and merges in sorted order.
- Handles remaining elements in either half.

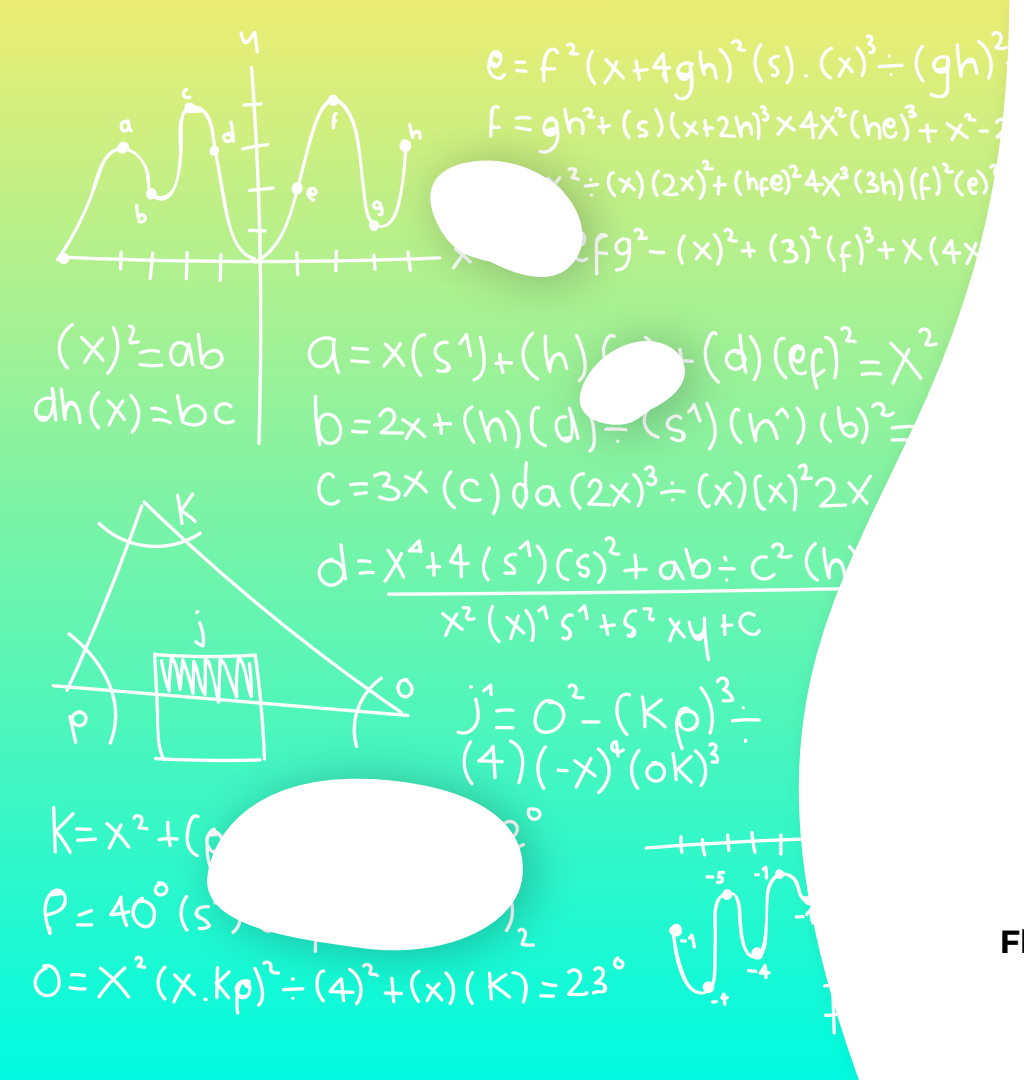
ΠΑΝΕΠΙΣΤΗΜΙΟ

Python File Generation

Purpose: Create random integers and store them in a binary file.
Sample Code:

```
for _ in range(1000):
    integers.append(random.randint(1, 1000))
write_integers_to_binary_file(file_name, integers)
```

Output: A binary file 'integers.bin' containing 1000 random integers.



Thanks!

Do you have any questions?

Evryviadis Liapis
Evaggelos Plytas
Aikaterini Tsiaousi