

COSC 3360 - 23466 - Operating Systems / COSC 6310 - 25945 - Fundamentals of Operating Systems


[Dashboard](#) / [My courses](#) / [COSC3360F202323466](#) / [PROGRAMMING ASSIGNMENTS](#) / [Programming Assignment 1](#)


 [Description](#)



 [Submission](#)

 [Edit](#)

 [Submission view](#)

 **Available from:** Monday, 21 August 2023, 12:00 AM

 **Due date:** Saturday, 23 September 2023, 11:59 PM

 **Requested files:** main.cpp ( [Download](#))

Type of work:  Individual work

This programming assignment closes at 11:58:59 PM on 09/23/2023.

Similarity Threshold: 95%

Problem:

For this assignment, you will create a multithreaded version of the incremental entropy algorithm proposed by Dr. Rincon in the paper "Entropy-based scheduling performance in real-time multiprocessor systems" (<https://ieeexplore.ieee.org/abstract/document/10089704>).

Given an input string representing the tasks executed in a processor, you must calculate the entropy of each scheduling instant using the incremental entropy algorithm proposed by Dr. Rincon.

For example, given the following string:

A 2 B 4 C 3 A 7

Where A, B, and C represent the tasks executed in the processor, and 2, 4, 3, and 7, represent the time instants that the tasks are executed in the processor. The task identifier is represented by one character, and the task execution time is represented by a positive integer value.

Given the previous example, the scheduling time instants are 0, 2, 6, and 9, and the entropy for scheduling time instants are:

$H(0) = 0$, $H(2) = 0.918295$, $H(6) = 1.530493$, and $H(9) = 1.419736$

Process:

Your solution must execute the following steps:

- Read from STDIN the n strings, where each string represents the scheduling information of a CPU in a multiprocessor platform.
- Create n POSIX threads (where n is the number of strings). Each child thread executes the following tasks:
 - Receives the string with the scheduling information of the assigned CPU from the main thread.
 - Uses the incremental entropy algorithm proposed by Dr. Rincon to calculate the entropy of the CPU at each

scheduling instant.

- Stores the calculated entropy values on a memory location accessible by the main thread.

- Print the information about the entropy values for all strings from the input.

Given the following input strings:

```
A 2 B 4 C 3 A 7
B 3 A 3 C 3 A 1 B 1 C 1
```

The expected output is:

```
CPU 1
Task scheduling information: A(2), B(4), C(3), A(7)
Entropy for CPU 1
0.00 0.92 1.53 1.42

CPU 2
Task scheduling information: B(3), A(3), C(3), A(1), B(1), C(1)
Entropy for CPU 2
0.00 1.00 1.58 1.57 1.57 1.58
```

NOTES:

- You can safely assume that the input will always be in the proper format.
- You must use the output statement format based on the example above.
- You can define additional functions if needed.
- You must present code that is readable and has comments explaining the logic of your solution. A 10% penalty will be applied to submissions not following this guideline.
- You cannot use global variables. A 100% penalty will be applied to submissions using global variables.
- Not using multiple POSIX threads to implement your solution will translate into a penalty of 100%.
- A penalty of 100% will be applied to solutions that do not compile.
- A penalty of 100% will be applied to solutions hardcoding the output.

ASSIGNMENT RUBRIC:

- Correct output: 5 points per test case (20 points total).
- Correct implementation of the incremental entropy algorithm: 20 points.
- Taking full advantage of multithreading (no pthread_join or sleep in the same loop as pthread_create): 30 points.
- Creating the correct number of threads: 20 points.
- Having clean and commented code: 10 points.

Penalties:

1. Presenting a solution that does not compile: -100 points.
2. Not using POSIX threads: -100 points.
3. Hardcoding the output: -100 points.
4. Using global variables: -100 points.

Requested files

main.cpp

```
1 // Write your program here
```