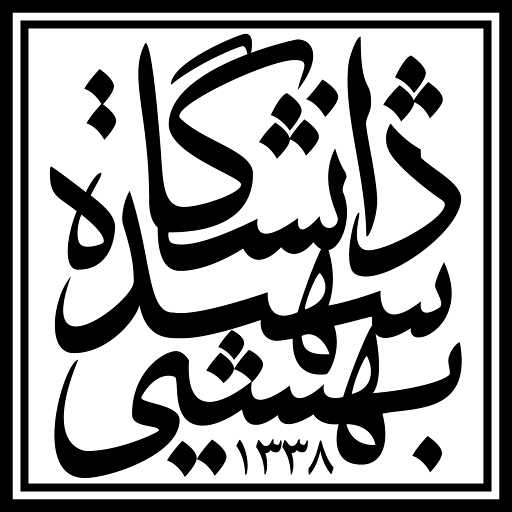
In The Name of God



Sixth Assignment Report

Amirhossein Donyaei

Student ID: 401222056

Spring 2023

Introduction:

This series of exercises will examine multi-Threading and its syntax in Java. One of the important features of programming languages, including Java, is multi-Threading which is used to improve performance and reduce speed in complex and large programs. Here we do some simple exercises to learn more about multi-Threading.

1.CPU-Simulator:

In this exercise, our goal is to learn the multi-threading syntax and build some threads. This was done with past learnings and with research. Another major challenge in this exercise is sorting tasks based on their processing time, which was done with the sortTask() function. Overall, it was an easy exercise to learn the syntax of multi-Thread in Java.

2.FindMultiples:

In this exercise, we will add the multiples of 3, 5, and 7 up to the number n, and we will use the principle of inclusion and exclusion. Using the multi-thread, the multiple of each number is calculated separately and added to the sum variable. This creates race conditioning. To solve this problem, there are ways that it was preferred to use lock class.

3.UseInterrupts:

In the third exercise, we have to interrupt threads with a time of more than three seconds. in this exercise interrupt() method in the Thread class helps us. with the interrupt() method, we tell to the thread to end itself. then we just handle how to end the thread when became interrupted.

4.CPU-Simulator Dual Core (Bonus Objective 1):

Not much difference from the first exercise! We just added another core to the program. Now we send tasks to the cores based on lower process time Respectively. then, when a core ended a task, we send another task to it.

5.FindMultiples Thead Pool (Bonus Objective 2)

This exercise also is not much different from the second exercise. In this exercise, we will learn how to use a thread pool and learn its syntax in Java.

What is thread pool?

Thread pool is a way to manage and use threads more efficiently in Java. In a thread pool, a group of threads wait for asynchronous tasks to be assigned to them. When a thread is assigned a task, it completes it and returns to the thread pool to wait for the next task.

The main difference between thread pool and ordinary threads is in the number of threads created to execute tasks. In ordinary threads, a new thread is created for each task, and if another task needs to be executed, another thread must be created. In thread pool, a limited number of threads are created to execute all tasks. This results in fewer threads being created and better performance for the program.

generally, thread pool is a powerful feature in Java that allows multiple threads to be managed and used efficiently. It is a way of creating efficient and responsive applications by dividing the program into smaller, independent threads that can run simultaneously. The main advantage of thread pool over ordinary threads is that it creates a group of threads to execute tasks, which results in better performance and fewer threads being created.

Conclusion:

In this assignment, we did practical and relatively simple exercises to get familiar with multi-threading and its syntax in Java language. Also, we learned other factors such as how to deal with race conditions, how to work with the thread pool, etc.