**Assignment 3 Documentation**

**Problem 1**

The Minotaur’s birthday party was a success. The Minotaur received a lot of presents from his guests. The next day he decided to sort all of his presents and start writing “Thank you” cards. Every present had a tag with a unique number that was associated with the guest who gave it. Initially all of the presents were thrown into a large bag with no particular order. The Minotaur wanted to take the presents from this unordered bag and create a chain of presents hooked to each other with special links (similar to storing elements in a linked-list). In this chain (linked-list) all of the presents had to be ordered according to their tag numbers in increasing order. The Minotaur asked 4 of his servants to help him with creating the chain of presents and writing the cards to his guests. Each servant would do one of three actions in no particular order:

1. Take a present from the unordered bag and add it to the chain in the correct location by hooking it to the predecessor’s link. The servant also had to make sure that the newly added present is also linked with the next present in the chain.

2. Write a “Thank you” card to a guest and remove the present from the chain. To do so, a servant had to unlink the gift from its predecessor and make sure to connect the predecessor’s link with the next gift in the chain.

3. Per the Minotaur’s request, check whether a gift with a particular tag was present in the chain or not; without adding or removing a new gift, a servant would scan through the chain and check whether a gift with a particular tag is already added to the ordered chain of gifts or not.

As the Minotaur was impatient to get this task done quickly, he instructed his servants not to wait until all of the presents from the unordered bag are placed in the chain of linked and ordered presents. Instead, every servant was asked to alternate adding gifts to the ordered chain and writing “Thank you” cards. The servants were asked not to stop or even take a break until the task of writing cards to all of the Minotaur’s guests was complete. After spending an entire day on this

task the bag of unordered presents and the chain of ordered presents were both finally empty! Unfortunately, the servants realized at the end of the day that they had more presents than “Thank you” notes. What could have gone wrong? Can we help the Minotaur and his servants improve their strategy for writing “Thank you” notes? Design and implement a concurrent linked-list that can help the Minotaur’s 4 servants with this task. In your test, simulate this concurrent “Thank you” card writing scenario by dedicating 1 thread per servant and assuming that the Minotaur received 500,000 presents from his guests.

**Solution**

The chosen algorithm incorporates the concept of concurrent programming by dedicating multiple threads to operate simultaneously on shared data on the linked list. In this scenario, each servant is represented by a separate thread, allowing them to perform tasks concurrently without waiting for others to complete.

The Linked\_List class serves as the data structure to organize the presents in the desired order. It includes methods to add and remove presents while ensuring thread safety through mutex locks.

The Servant class defines the behavior of the individual servants. Each servant thread alternates between adding presents to the linked-list and writing "Thank you" notes. This alternating action ensures a continuous workflow without waiting for all presents to be processed before starting the note-writing task.

**1. Initialization:**

- First we create a Linked\_List object to represent the chain of presents.

- Instantiate four Servant objects, each represented by a servant thread.

**2. Thread Execution:**

- Start each servant thread concurrently.

- Each servant thread executes a loop to perform the following actions:

- Add a present to the linked-list.

- Remove a present from the linked-list and write a "Thank you" note.

- Repeat the above steps until all presents are processed.

**3. Adding Presents:**

- Each servant thread adds presents to the linked-list in the correct order.

- The linked-list ensures that presents are inserted in increasing order based on their tag numbers.

- Mutex locks are used to ensure thread safety during the addition of presents.

**4. Writing Thank You Notes:**

- After adding a present, each servant thread removes a present from the linked-list to write a "Thank you" note.

- The present is unlinked from the linked-list, ensuring that the chain remains intact.

The algorithm demonstrates correctness through its concurrent execution and continuous operation, this means that each worker is allowed to work independently of the others and in their own timeline. By allowing all four servant threads to operate independently and concurrently, the algorithm maximizes CPU utilization while ensuring thread safety through mutex locks during present addition and removal. This guarantees efficient processing of presents and timely note writing. Upon completion, the linked-list empties, indicating successful present sorting and note writing. This comprehensive approach, coupled with output messages for progress tracking, validates the algorithm's correctness in accurately managing the Minotaur's gift sorting and "Thank you" note writing tasks.

Below are a few runtime results for the program:

```

Servant 3 added 125000 presents and wrote 125000 'thank you' notes.

Servant 1 added 125000 presents and wrote 125000 'thank you' notes

Servant 2 added 125000 presents and wrote 125000 'thank you' notes

Servant 0 added 125000 presents and wrote 125000 'thank you' notes

Time taken: 0.107s

```

```

Servant 0 added 125000 presents and wrote 125000 'thank you' notes

Servant 3 added 125000 presents and wrote 125000 'thank you' notes

Servant 2 added 125000 presents and wrote 125000 'thank you' notes

Servant 1 added 125000 presents and wrote 125000 'thank you' notes

Time taken: 0.11s

```

with double the servants: 8

```

Servant 2 added 62500 presents and wrote 62500 'thank you' notes

Servant 5 added 62500 presents and wrote 62500 'thank you' notes

Servant 7 added 62500 presents and wrote 62500 'thank you' notes

Servant 3 added 62500 presents and wrote 62500 'thank you' notes

Servant 6 added 62500 presents and wrote 62500 'thank you' notes

Servant 4 added 62500 presents and wrote 62500 'thank you' notes

Servant 1 added 62500 presents and wrote 62500 'thank you' notes

Servant 0 added 62500 presents and wrote 62500 'thank you' notes

Time taken: 0.166s

```

```

**Problem 2**

You are tasked with the design of the module responsible for measuring the atmospheric

temperature of the next generation Mars Rover, equipped with a multicore CPU and 8

temperature sensors. The sensors are responsible for collecting temperature readings at

regular intervals and storing them in shared memory space. The atmospheric

temperature module has to compile a report at the end of every hour, comprising the top

5 highest temperatures recorded for that hour, the top 5 lowest temperatures recorded

for that hour, and the 10-minute interval of time when the largest temperature

difference was observed. The data storage and retrieval of the shared memory region

must be carefully handled, as we do not want to delay a sensor and miss the interval of

time when it is supposed to conduct temperature reading. Design and implement a

solution using 8 threads that will offer a solution for this task. Assume that the

temperature readings are taken every 1 minute. In your solution, simulate the operation

of the temperature reading sensor by generating a random number from -100F to 70F at

every reading. In your report, discuss the efficiency, correctness, and progress guarantee

of your program.

**Solution**

For this problem I used a random generator to simulate the temperature reading from the sensors. Each sensor is assigned to a thread of its own.The main function simulates the temperature readings for a whole day.To maintain the highest and lowest temperatures in each hour plus recording the largest temperature reading, I used the priority queue for easier computing, and retrieval of data.

- The min heap (min\_heap) is used to store the top 5 highest temperatures. When the size of the heap is less than or equal to 5, you simply push the new temperature onto the heap. If the size exceeds 5 and the new temperature is greater than the current top element (which is the smallest of the top 5 highest temperatures), you replace the current top element with the new temperature.

- The max heap (max\_heap) is used to store the top 5 lowest temperatures. When the size of the heap is less than or equal to 5, you push the new temperature onto the heap. If the size exceeds 5 and the new temperature is smaller than the current top element (which is the largest of the top 5 lowest temperatures), you replace the current top element with the new temperature.

- The largest difference is simply(max temp of the hour - min temp of the hour) which are top of the stacks.

This algorithm simulates the collection of temperature readings from multiple sensors over the course of 24:

**1.Initialization:**

- The main function initializes variables such as the number of sensors, number of intervals, and the duration in hours. It also sets up a vector of threads to handle concurrent sensor readings.

**2. Sensor Reading Simulation:**

- Within a loop iterating over each hour of the day, another loop iterates over each sensor. For each sensor, a thread is created using the `sensor\_reading` function, which simulates reading temperatures for a specific number of intervals `sensors\_per\_interval`. These threads operate concurrently to simulate real-time sensor readings.

**3. Temperature Reading and Storage:**

- Inside the `sensor\_reading` function, a loop iterates for the specified number of intervals. For each interval, a random temperature reading within a predefined range is generated. These readings are then stored in two priority queues: `min\_heap` to maintain the top 5 highest temperatures and `max\_heap` to maintain the top 5 lowest temperatures. Additionally, the maximum temperature difference within each 10-minute interval is calculated and updated.

**4. Compilation of Report:**

- After all sensor readings for an hour are completed, the `compile\_report` function is called. This function acquires a lock before accessing shared data to ensure thread safety. It then prints out the top 5 highest and lowest temperatures recorded throughout the hour, followed by the maximum temperature difference observed during 10-minute intervals.

**5. Thread Management:**

- The `main()` function joins all sensor threads after completing readings for each hour, ensuring that all readings are processed before compiling the report. It also clears the `sensor\_threads` vector to prepare for the next hour's readings.

Here is a simulation of a whole day: 24hrs -;

```

Hour : 0

Top 5 Highest Temperatures: 45.5957 47.1599 50.0327 53.9121 59.9572 65.287

Top 5 Lowest Temperatures: -86.7955 -92.5499 -92.7213 -93.27 -94.5823 -96.5156

Maximum Temperature Difference at 10-minute Intervals: Interval: 21 - 30, Max Temp Difference: 132.391

Hour : 1

Top 5 Highest Temperatures: 61.8075 62.2025 62.8042 63.3081 65.287 66.7351

Top 5 Lowest Temperatures: -89.5515 -92.5499 -92.7213 -93.27 -94.5823 -96.5156

Maximum Temperature Difference at 10-minute Intervals: Interval: 1 - 10, Max Temp Difference: 151.359

Hour : 2

Top 5 Highest Temperatures: 62.8042 63.3081 65.287 66.7351 68.3077 69.4297

Top 5 Lowest Temperatures: -92.5499 -92.7213 -93.27 -94.5823 -96.3297 -96.5156

Maximum Temperature Difference at 10-minute Intervals: Interval: 21 - 30, Max Temp Difference: 155.354

Hour : 3

Top 5 Highest Temperatures: 65.287 66.7351 66.7815 67.2252 68.3077 69.4297

Top 5 Lowest Temperatures: -92.7213 -93.27 -94.386 -94.5823 -96.3297 -96.5156

Maximum Temperature Difference at 10-minute Intervals: Interval: 41 - 50, Max Temp Difference: 158.008

Hour : 4

Top 5 Highest Temperatures: 66.7351 66.7815 67.2252 68.2252 68.3077 69.4297

Top 5 Lowest Temperatures: -94.5823 -96.3297 -96.5156 -97.3309 -97.5041 -99.1763

Maximum Temperature Difference at 10-minute Intervals: Interval: 21 - 30, Max Temp Difference: 161.317

Hour : 5

Top 5 Highest Temperatures: 66.7815 67.2252 67.2586 68.2252 68.3077 69.4297

Top 5 Lowest Temperatures: -96.5156 -97.3309 -97.5041 -97.6533 -99.1763 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: 41 - 50, Max Temp Difference: 163.297

Hour : 6

Top 5 Highest Temperatures: 66.7815 67.2252 67.2586 68.2252 68.3077 69.4297

Top 5 Lowest Temperatures: -97.3627 -97.4072 -97.5041 -97.6533 -99.1763 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: 41 - 50, Max Temp Difference: 164.144

Hour : 7

Top 5 Highest Temperatures: 67.2252 67.2586 68.2252 68.3077 68.5404 69.4297

Top 5 Lowest Temperatures: -97.4072 -97.5041 -97.6533 -99.1387 -99.1763 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: 21 - 30, Max Temp Difference: 164.632

Hour : 8

Top 5 Highest Temperatures: 67.2586 67.564 68.2252 68.3077 68.5404 69.4297

Top 5 Lowest Temperatures: -97.4072 -97.5041 -97.6533 -99.1387 -99.1763 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: 11 - 20, Max Temp Difference: 164.666

Hour : 9

Top 5 Highest Temperatures: 67.564 68.2252 68.238 68.3077 68.5404 69.4297

Top 5 Lowest Temperatures: -97.4072 -97.5041 -97.6533 -99.1387 -99.1763 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: 41 - 50, Max Temp Difference: 164.971

Hour : 10

Top 5 Highest Temperatures: 67.564 68.2252 68.238 68.3077 68.5404 69.4297

Top 5 Lowest Temperatures: -97.4822 -97.5041 -97.6533 -99.1387 -99.1763 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: -9 - 0, Max Temp Difference: 165.046

Hour : 11

Top 5 Highest Temperatures: 68.2252 68.238 68.3077 68.5404 69.4297 69.4627

Top 5 Lowest Temperatures: -97.6533 -98.0862 -99.1387 -99.1763 -99.5501 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: 31 - 40, Max Temp Difference: 165.879

Hour : 12

Top 5 Highest Temperatures: 68.2252 68.238 68.3077 68.5404 69.4297 69.4627

Top 5 Lowest Temperatures: -98.0862 -98.4675 -99.1387 -99.1763 -99.5501 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: 41 - 50, Max Temp Difference: 166.311

Hour : 13

Top 5 Highest Temperatures: 68.238 68.3077 68.5404 69.4297 69.4627 69.5786

Top 5 Lowest Temperatures: -98.4026 -98.4675 -99.1387 -99.1763 -99.5501 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: 21 - 30, Max Temp Difference: 166.641

Hour : 14

Top 5 Highest Temperatures: 68.238 68.3077 68.5404 69.4297 69.4627 69.5786

Top 5 Lowest Temperatures: -98.4026 -98.4675 -99.1387 -99.1763 -99.5501 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: 21 - 30, Max Temp Difference: 166.641

Hour : 15

Top 5 Highest Temperatures: 68.5404 69.1868 69.4048 69.4297 69.4627 69.5786

Top 5 Lowest Temperatures: -98.489 -98.5447 -99.1387 -99.1763 -99.5501 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: 11 - 20, Max Temp Difference: 167.029

Hour : 16

Top 5 Highest Temperatures: 69.1868 69.4048 69.4297 69.4627 69.5786 69.6542

Top 5 Lowest Temperatures: -98.489 -98.5447 -99.1387 -99.1763 -99.5501 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: 31 - 40, Max Temp Difference: 167.676

Hour : 17

Top 5 Highest Temperatures: 69.1868 69.4048 69.4297 69.4627 69.5786 69.6542

Top 5 Lowest Temperatures: -98.489 -98.5447 -99.1387 -99.1763 -99.5501 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: 31 - 40, Max Temp Difference: 167.676

Hour : 18

Top 5 Highest Temperatures: 69.1868 69.4048 69.4297 69.4627 69.5786 69.6542

Top 5 Lowest Temperatures: -98.489 -98.5447 -99.1387 -99.1763 -99.5501 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: 31 - 40, Max Temp Difference: 167.676

Hour : 19

Top 5 Highest Temperatures: 69.2053 69.4048 69.4297 69.4627 69.5786 69.6542

Top 5 Lowest Temperatures: -98.489 -98.5447 -99.1387 -99.1763 -99.5501 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: 51 - 60, Max Temp Difference: 167.694

Hour : 20

Top 5 Highest Temperatures: 69.4048 69.4297 69.4627 69.5786 69.6542 69.7737

Top 5 Lowest Temperatures: -98.5447 -98.9893 -99.1387 -99.1763 -99.5501 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: 11 - 20, Max Temp Difference: 167.95

Hour : 21

Top 5 Highest Temperatures: 69.4048 69.4297 69.4627 69.5786 69.6542 69.7737

Top 5 Lowest Temperatures: -98.9893 -99.1387 -99.1763 -99.4346 -99.5501 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: -9 - 0, Max Temp Difference: 168.394

Hour : 22

Top 5 Highest Temperatures: 69.4048 69.4297 69.4627 69.5786 69.6542 69.7737

Top 5 Lowest Temperatures: -98.9893 -99.1387 -99.1763 -99.4346 -99.5501 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: -9 - 0, Max Temp Difference: 168.394

Hour : 23

Top 5 Highest Temperatures: 69.4297 69.4627 69.5786 69.6542 69.7737 69.9928

Top 5 Lowest Temperatures: -99.1579 -99.1763 -99.4346 -99.5501 -99.592 -99.9052

Maximum Temperature Difference at 10-minute Intervals: Interval: 41 - 50, Max Temp Difference: 168.588

Execution Time: 32167 Milliseconds

```