Name: Vu Hoang Tuan Anh
Class: CS 146.
Project 2 Report

ReadFile.java

- **ReadFile():** Constructor.
- **ArrayList<Integer[]> readFile ():** readFile method use Scanner to read every line in sumTest.txt. Then separate each value on a line and Store into ArrayList.

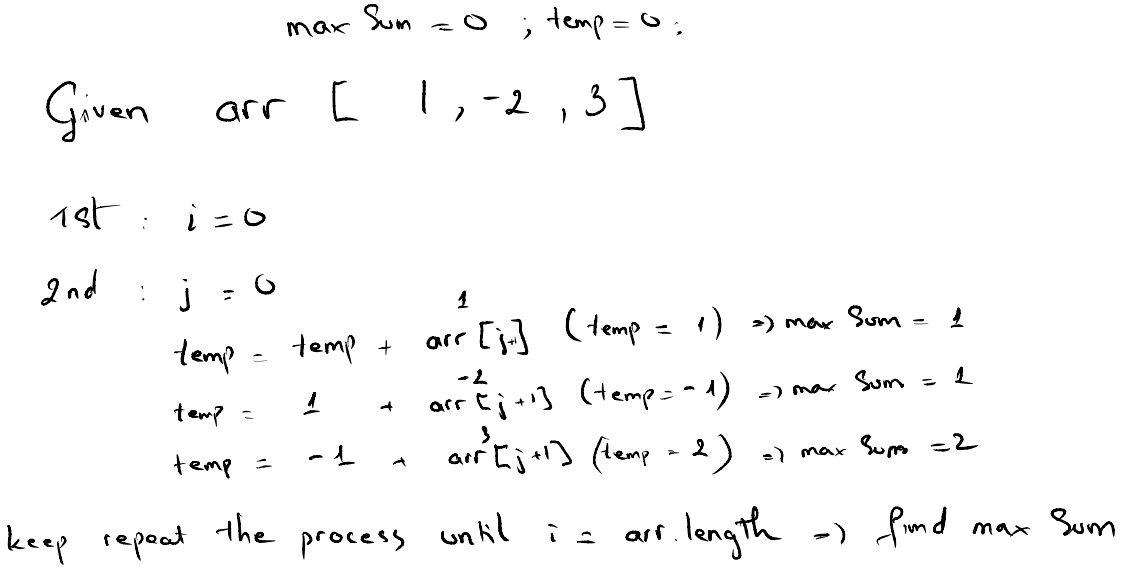
RunningTime.java

- **RunningTime ():** Constructor, run getRandomArr() method whenever it is called.
- **ArrayList<Integer[]> getRandomArr():** return an array of random number and store into an ArrayList.
- **float [] getRunningTime:** measuring the time find Max Sum of each Integer Array ten times, then get the average of the time running from each method.

WorstMethod.java: **Time Complexity $O(n^2)$**

- **WorstMethod(Integer[] arrTest):** Constructor pass array value from user.
- **subSumArr():** Find the max Sum of an existed Array by using two for loop.

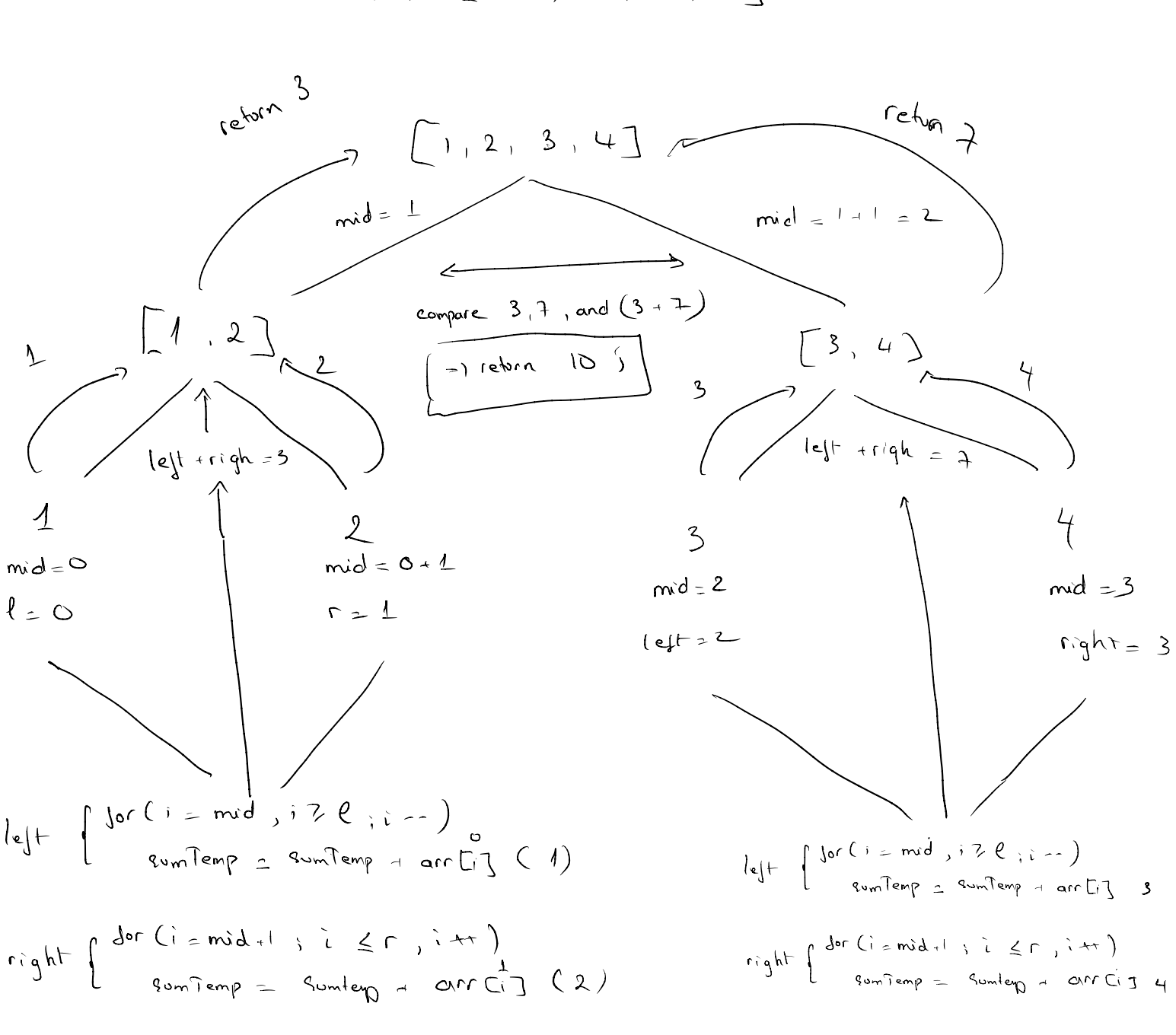
How it works?



AverageMethod.java: **Time Complexity O(nlgn)**

- **AverageMethod(Integer[] arrTest):** Constructor pass array value from user.
- **subSumArr (Integer[] arr, int l, int r):** Get value from user. set l = 0, and r = arr.length - 1). subSumArr follow Algorithm D&C use divide and Conquer approach. Keep divide array into half, then find the max sum from left half and right half. Then find the max sum in the middle by combine the left and the right half together. Finally, compare all max Sum from left, right, and middle to find the max Sum.

How it works?



BestMethod.java: **Time Complexity O(n)**

- **BestMethod(Integer[] arrTest):** Constructor pass array value from user.
- **subSumArr():** Find the max Sum of an existed Array by using Kadane's Algorithm.

How it works?

Arr [1, 2, -3, 4]

int maxTemp = 0

int maxSum = 0

loop through Arr [1, 2, -3, 4]

loop 1 : maxTemp = maxTemp + Arr[0] (1)

maxSum = 1

loop 2 : maxTemp = 1 + arr(1) (3)

maxSum = 3

loop 3 : maxTemp = 3 + arr(2) (0)

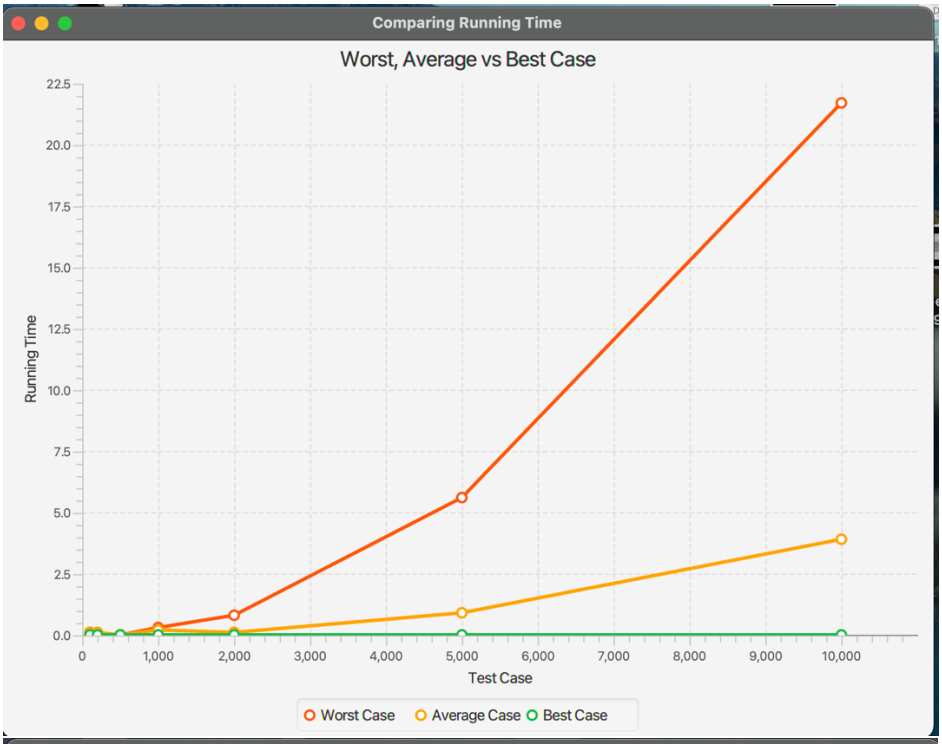
keep maxSum = 3

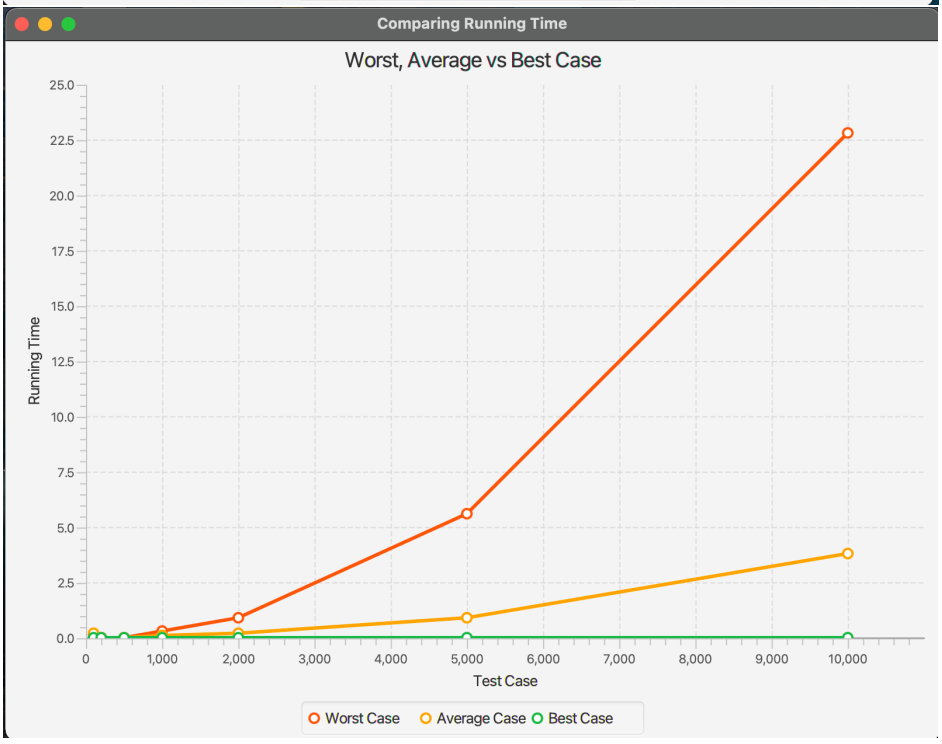
loop 4 : maxTemp = 0 + arr(3) (4)

maxTemp > maxSum => maxSum = 4

Always reset maxTemp whenever maxTemp < 0.

Line Chart


Comparing Running Time
Worst, Average vs Best Case
Running Time
22.5
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0
1,000
2,000
3,000
4,000
5,000
6,000
7,000
8,000
9,000
10,000
Test Case
Worst Case
Average Case
Best Case


Comparing Running Time
Worst, Average vs Best Case
Running Time
25.0
22.5
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0
1,000
2,000
3,000
4,000
5,000
6,000
7,000
8,000
9,000
10,000
Test Case
Worst Case
Average Case
Best Case