Project 2

Problem 1: Optimization of production lines

Pseudo code:

Define Production\_Optimization (Durations[], stations: int):

#for this function we will use the greedy approach to optimize the production line.

#we will define the lower bound and the higher bound to find the longest station

#the longest duration of the single station can be between the longest\_durations of a single station to the total summation of all durations in the Durations[] array.

Lower\_bound = maximum value of Durations[]

Higher\_bound = Summation of all durations in Durations[]

#define the result to return the longest duration

Result = max(“Inf”)

#Optimize using binary search. In between the lower bound and the higher bound, we calculate the longest\_estimate\_value to be the middle of the lower bound and higher bound.

While lower\_bound <= higher\_bound:

Middle = lower\_bound + ((higher\_bound – lower\_bound) // 2)

#we check if we can divide stations further by estimating if the can\_divide() is in the lower half or the upper half of values between the lower\_bound and the higher\_bound.

If can\_divide(middle): #if we can continue to divide then the solution is in the lower half , we reduce the higher\_bound by half

Result = middle

Higher\_bound = middle – 1

Else: #if not we increase the lower\_bound because we know the solution will be in the upper half between the lower bound and the higher bound.

Lower\_bound

Return result

#checking if the line can be optimized. A line can be optimized if the duration of the current station is less than the longest\_estimate\_value.

Define can\_divide(longest\_estimate\_value):

number\_Of\_station = 0 #initiate the number of stations

Duration\_of\_current\_station = 0 #initiate the duration of the current station

#traversing through the Durations[] and adding the duration to 1 station until we cannot add the duration (duration is greater than the longest\_estimate\_value)

For duration in durations[]:

Duration\_of\_current\_station = Duration\_of\_current\_station + duration

If (duration\_of\_current\_station > longest\_estimate\_value):

number\_of\_station = number\_of\_station + 1 #we increment the number\_of\_station to indicate this is the longest duration we could estimate for 1 station and continue to the next station.

Duration\_of\_current\_station = duration #when duration\_of\_current\_station cannot be added due to exceeding the estimated value, we start the next station with duration.

#after greedily separating the stations, we want to return to the function can\_divide by checking if the number\_of\_station is greater than our input stations.

return number\_of\_status + 1 <= stations

Mathematic prove: