Project 4.

To aid with testing a random number generator will be used to generate random floors to which the elevator must travel to.

The elevator will start on the first floor of the building.

The elevator will search through the list of floors inputted and determine which floor is closest to its current position.

It will repeat this step until there are no more floors left to travel to.

The elevator will display its current floor, where it is going, and what floors it passes on the way there.

A timer will be used to time how long the elevator takes to travel to every inputted floor.

This is an example when there are three level requests.

Diagram:

Level requests: 5,1,6A picture containing diagram

Description automatically generatedA picture containing text, shoji, handcart

Description automatically generated

The elevator starts at floor 1.

Floor 5 is 4 floors away.

Floor 6 is 5 floors away.

Floor 5 is closer.

The elevator will travel to floor 5 first.

Icon

Description automatically generatedIcon

Description automatically generatedSo the order will be.

Icon

Description automatically generated

ADL:

declare max, numberOfFloors, currentLevel, levelList[], lvl[]

for i <- 1 to max by 1

declare random

random <- CALL random(numberOfFloors)

levelList[i]<-random

end

print(“The elevator will go to these floors: “+levelList)

for i2 <- 1 to levelList by 1

declare diff

declare level

diff<-levelList[i2]-currentLevel

level<-(diff,levelList[i2])

lvl[i]<-level

end

CALL sort(lvl,CALL CompareLevel)

print(“The current floor is: “+currentLevel)

declare timeStart

timeStart <- CALL getTime()

while (!lvl<-CALL Empty())

declare level

level <- lvl[0]

print(“level request “+level)

while (level>currentLevel)

print(currentLevel)

currentLevel by 1

end

while(level<currentLevel)

print(currentLevel)

currentLevel down by 1

end

print(“You have arrived at level “+currentLevel)

lvl[0]<-CALL remove

end

declare endTime, time

endTime <- CALL getTime()

time <- endTime-timeStart

print(“Duration : “+time+” seconds”)

Testing results:



Descriptive Report:

Experimental Strategy

My testing strategy was to run empirical experiments for both datasets. I tested both algorithms against 7 different level requests. As the number of level requests increased, I also increased the number of floors available to be travelled to. This improved testing as the same floor was less likely to be revisited. To further improve accuracy I tested 10 samples for each number of level requests and calculated the mean and minimum values. I then plotted a graph of these mean and minimum values to demonstrate the efficiency of each algorithm.

Experiment Results

The results demonstrated in the graph show that the mean and minimum for Michael’s Algorithm have a linear runtime for all values of N. The results also show that the mean and minimum for FIFO Algorithm have an exponential runtime for values of N. There is some inconsistency in the graph this is because not every single value of N was tested.

Findings and Discussion

It is evident that as the number of floors the elevator needs to visit increase the time it takes to finish its journey increases linearly rather than exponentially like the FIFO algorithm, as evident in the graph. This is because the algorithm is more efficient than a First In First Out approach as floors are sorted by their distance from the starting point of the elevator. This means that the elevator will not miss a floor it needs to stop at, resulting in the elevator travelling back to that floor therefore the elevator only must traverse the length of the building 1 time. Resulting in a much lower runtime as the number of floors (N) increase.

Video:

<https://youtu.be/A0q4xAFerPo>

Formative Feedback:

Action Plan

|  |  |  |
| --- | --- | --- |
| Feedback | Implement Changes | Have the changes been implemented |
| It’s good to have a diagram too to illustrate your algorithm. This part is suitable to be illustrated in a diagram. “The elevator will search through the list of floors inputted and determine which floor is closest to its current position”. | Add a diagram illustrating the algorithm deciding what order to travel to each floor. | Yes |
| Attach all the results (for every Ns - your excel file) in the report. | Paste a link to the excel spreadsheet in your portfolio file. | Yes |
| Plot this table into a line chart to see comparison between algorithms. | Use excel to make a line chart for the Mean and Min values of each algorithm. | Yes |
| Use the report template. There is a section findings and discussion. This section should have detailed discussion about the results and graph. | Implement the report template and the graph to finish with a more statistical conclusion. | Yes |