Project 1 Output without the deliberate throw of an exception

run:

O(1) method calculational verification

Trial 1

n= 1 time/1= 1399.0

n= 10 time/1= 350.0

n= 100 time/1= 350.0

n= 1000 time/1= 0.0

n= 10000 time/1= 0.0

n= 100000 time/1= 350.0

n= 1000000 time/1= 350.0

n= 10000000 time/1= 0.0

Trial 2

n= 1 time/1= 1399.0

n= 10 time/1= 350.0

n= 100 time/1= 0.0

n= 1000 time/1= 350.0

n= 10000 time/1= 350.0

n= 100000 time/1= 0.0

n= 1000000 time/1= 350.0

n= 10000000 time/1= 0.0

Since time/constant is not growing with n demoBigO1 is O(1)

O(1) is not O(logn) method calculational verification

Trial 1

n= 1 time/Log(n)= NaN

n= 10 time/Log(n)= 1216.024549329105

n= 100 time/Log(n)= 151.7859214251865

n= 1000 time/Log(n)= 0.0

n= 10000 time/Log(n)= 38.00076716653453

n= 100000 time/Log(n)= 0.0

n= 1000000 time/Log(n)= 0.0

n= 10000000 time/Log(n)= 0.0

Trial 2

n= 1 time/Log(n)= Infinity

n= 10 time/Log(n)= 152.00306866613812

n= 100 time/Log(n)= 76.00153433306906

n= 1000 time/Log(n)= 0.0

n= 10000 time/Log(n)= 0.0

n= 100000 time/Log(n)= 30.400613733227626

n= 1000000 time/Log(n)= 0.0

n= 10000000 time/Log(n)= 0.0

Since time/Log(n) is shrinking too rapidly with n demoBigO1 is worstTime(n) less than O(Log(n))

Editor’s note: This is not very convincing data, but it is easy to see a O(1) algorithm. It is much more important to properly classify O(nlog(n)) and O(n\*n) algorithms.

O( Log(n) ) method calculational verification

n= 1 time/Log(n)= Infinity

n= 10 time/Log(n)= 152.00306866613812

n= 100 time/Log(n)= 152.00306866613812

n= 1000 time/Log(n)= 101.33537911075877

n= 10000 time/Log(n)= 76.00153433306906

n= 100000 time/Log(n)= 60.80122746645525

n= 1000000 time/Log(n)= 76.00153433306907

Since time/log(n) is not growing with n demoBigOLogN is O(log n)

O(n) method calculational verification

n= 1 time/n= 2799.0

n= 10 time/n= 140.0

n= 100 time/n= 27.99

n= 1000 time/n= 20.647

n= 10000 time/n= 23.621

n= 100000 time/n= 26.04251

n= 1000000 time/n= 7.559395

Since time/n is not growing with n demoBigOLogN is O(n)

O(n) is BIGGER than and not O( log(n) ) method calculational verification

n= 1 time/Log(n)= Infinity

n= 10 time/Log(n)= 152.00306866613812

n= 100 time/Log(n)= 76.00153433306906

n= 1000 time/Log(n)= 354.67382688765565

n= 10000 time/Log(n)= 2621.6186400089796

n= 100000 time/Log(n)= 21945.50818284598

n= 1000000 time/Log(n)= 144707.28328223177

Since time/Log(n) is growing with n demoBigOn is worstTime(n) bigger and worse than O(Log(n))

O(n) is not O(n log(n) ) method calculational verification

n= 1 time/nLog(n)= Infinity

n= 10 time/nLog(n)= 91.20184119968287

n= 100 time/nLog(n)= 8.35799730422808

n= 1000 time/nLog(n)= 3.799352892516948

n= 10000 time/nLog(n)= 2.591207168913704

n= 100000 time/nLog(n)= 2.189985514690831

n= 1000000 time/nLog(n)= 0.18979370968584516

Since time/nLog(n) is shrinking too rapidly with n demoBigOn is worstTime(n) less than O(nLog(n))

O( nLog(n) ) method calculational verification

n= 10 time/n\*log(n)= 29650.804310221658

n= 100 time/n\*log(n)= 594.2321107537623

n= 1000 time/n\*log(n)= 444.3818398344187

n= 10000 time/n\*log(n)= 190.38964133567143

n= 100000 time/n\*log(n)= 32.226193171221006

check this range in more detail

n= 100000 time/n\*log(n)= 33.87514851778018

n= 200000 time/n\*log(n)= 39.34345296362449

n= 300000 time/n\*log(n)= 50.641352847640086

n= 400000 time/n\*log(n)= 62.50999350124314

n= 500000 time/n\*log(n)= 53.90182177228439

n= 600000 time/n\*log(n)= 45.39953003928318

n= 700000 time/n\*log(n)= 57.4429904551785

n= 800000 time/n\*log(n)= 47.3101517885284

n= 900000 time/n\*log(n)= 51.768866296450994

n= 1000000 time/n\*log(n)= 56.38555439059978

n= 1100000 time/n\*log(n)= 106.48642074569742

n= 1200000 time/n\*log(n)= 49.7758335686898

n= 1300000 time/n\*log(n)= 54.03095831483726

Since time/n\*Log(n) is not growing with n demoBigONLogN is O(n)

O(n\*n) method calculational verification

n= 1 time/n\*n= 8748.0

n= 10 time/n\*n= 27.99

n= 100 time/n\*n= 24.2508

n= 1000 time/n\*n= 3.70516

n= 10000 time/n\*n= 0.94031571

n= 100000 time/n\*n= 0.9058009488

Since time/n\*n is not growing with n demoBigOnn is O(n\*n)

BUILD SUCCESSFUL (total time: 9 seconds)