Frederico Castro Banner N03139680
Homework 3 – Estimate Average time of merge sort

```
class Array
  def swap!(i,j)
    self[i], self[j] = self[j], self[i]
    self
  end
end

def generateRandomPermutation(n)
  sequence = []

  n.times do |i|
    sequence[i] = i
  end

  (n - 1).downto(1) do | index |
    randomIndex = rand(1..index)
    sequence.swap!(index, randomIndex)
  end

  sequence
end
def mergeSort(list)
  return list if list.size <= 1
  middle = list.size / 2
  left  = list[0, middle]
  right = list[middle, list.size - middle]
  merge(mergeSort(left), mergeSort(right))
end

def merge(left, right)
  temp = []
  until left.empty? or right.empty?
    if left.first <= right.first
      temp << left.shift
    else
      temp << right.shift
    end
    @comparisonsCounter += 1
  end
  temp.concat(left).concat(right)
end

def estimateAverageRuntime(xSort)
  averages = []
  (1000..10000).step(100) do |n|
    total = 0
    50.times do |i|
      @comparisonsCounter = 0
      randomList = generateRandomPermutation(n)
      send(xSort, randomList)
      total += @comparisonsCounter
    end
    averageComparisons = total.to_f/50
    averages << [ n, averageComparisons ]
    p n
  end
  averages
end

def prepareResultsToBeExported(results)
  preparedResults = []
  results.each do |result|
    n = result[0]
    average = result[1].to_f
    preparedResults <<  [ n, average, average/n, average/
(n*n), average/(n*Math.log2(n))]
  end
  preparedResults
end

def exportCsv(preparedResults, fileName)
  require 'csv'

  CSV.open("#{fileName}.csv", "w") do |csv|
    csv << ["n", "av", "av/n", "av/n^2", "av/nlogn"]
    preparedResults.each do | result |
      csv << result
    end
  end
end

def mainCall
  begin
    mergeSortAverages =
estimateAverageRuntime("mergeSort")
    mergeSortResults =
prepareResultsToBeExported(mergeSortAverages)
    exportCsv(mergeSortResults, "mergeSort")
    p "Done"
  rescue
    p "Error"
  end
end

mainCall()
```

MERGE SORT

n	av	av/n	av/n^2	av/nlogn
1000	8707	8.71	0.0087068	0.873673
1100	9740	8.85	0.0080500	0.876444
1200	10781	8.98	0.0074866	0.878295
1300	11836	9.10	0.0070037	0.880173
1400	12900	9.21	0.0065818	0.881671
1500	13960	9.31	0.0062044	0.882080
1600	15039	9.40	0.0058745	0.883058
1700	16128	9.49	0.0055805	0.884039
1800	17207	9.56	0.0053109	0.884015
1900	18311	9.64	0.0050722	0.884820
2000	19411	9.71	0.0048527	0.885066
2100	20534	9.78	0.0046562	0.885989
2200	21674	9.85	0.0044781	0.887289
2300	22821	9.92	0.0043140	0.888487
2400	23965	9.99	0.0041605	0.889256
2500	25116	10.05	0.0040185	0.890019
2600	26274	10.11	0.0038867	0.890799
2700	27432	10.16	0.0037630	0.891331
2800	28589	10.21	0.0036466	0.891647
2900	29753	10.26	0.0035378	0.891991
3000	30926	10.31	0.0034362	0.892457
3100	32095	10.35	0.0033398	0.892680
3200	33276	10.40	0.0032496	0.893077
3300	34461	10.44	0.0031644	0.893427
3400	35661	10.49	0.0030849	0.894060
3500	36843	10.53	0.0030076	0.894112
3600	38031	10.56	0.0029345	0.894225
3700	39231	10.60	0.0028657	0.894526
3800	40423	10.64	0.0027994	0.894532
3900	41633	10.68	0.0027372	0.894861
4000	42834	10.71	0.0026771	0.894921
4100	44021	10.74	0.0026187	0.894634
4200	45266	10.78	0.0025661	0.895441
4300	46512	10.82	0.0025155	0.896161
4400	47743	10.85	0.0024661	0.896506
4500	48990	10.89	0.0024193	0.897076
4600	50236	10.92	0.0023741	0.897558
4700	51470	10.95	0.0023300	0.897748
4800	52721	10.98	0.0022883	0.898176
4900	53974	11.02	0.0022480	0.898570
5000	55220	11.04	0.0022088	0.898783
5100	56484	11.08	0.0021716	0.899241
5200	57728	11.10	0.0021349	0.899319
5300	58992	11.13	0.0021001	0.899670
5400	60251	11.16	0.0020662	0.899896
5500	61519	11.19	0.0020337	0.900200
5600	62774	11.21	0.0020017	0.900284
5700	64042	11.24	0.0019711	0.900510
5800	65303	11.26	0.0019412	0.900602
5900	66574	11.28	0.0019125	0.900795
6000	67835	11.31	0.0018843	0.900807
6100	69101	11.33	0.0018571	0.900871
6200	70386	11.35	0.0018311	0.901143
6300	71648	11.37	0.0018052	0.901078
6400	72944	11.40	0.0017809	0.901421
6500	74237	11.42	0.0017571	0.901698
6600	75539	11.45	0.0017341	0.902040
6700	76805	11.46	0.0017109	0.901920
6800	78101	11.49	0.0016890	0.902142
6900	79398	11.51	0.0016677	0.902342
7000	80671	11.52	0.0016463	0.902239
7100	81968	11.54	0.0016260	0.902391
7200	83243	11.56	0.0016058	0.902272
7300	84554	11.58	0.0015867	0.902527
7400	85866	11.60	0.0015680	0.902764
7500	87147	11.62	0.0015493	0.902652
7600	88441	11.64	0.0015312	0.902663
7700	89749	11.66	0.0015137	0.902795
7800	91043	11.67	0.0014964	0.902768
7900	92337	11.69	0.0014795	0.902735
8000	93646	11.71	0.0014632	0.902822
8100	94946	11.72	0.0014471	0.902800
8200	96252	11.74	0.0014315	0.902828
8300	97590	11.76	0.0014166	0.903132
8400	98932	11.78	0.0014021	0.903455
8500	100268	11.80	0.0013878	0.903706
8600	101602	11.81	0.0013737	0.903907
8700	102957	11.83	0.0013602	0.904277
8800	104293	11.85	0.0013468	0.904467
8900	105637	11.87	0.0013336	0.904704
9000	106963	11.88	0.0013205	0.904767
9100	108329	11.90	0.0013082	0.905154
9200	109663	11.92	0.0012956	0.905259
9300	111016	11.94	0.0012836	0.905500
9400	112351	11.95	0.0012715	0.905582
9500	113695	11.97	0.0012598	0.905718
9600	115051	11.98	0.0012484	0.905939
9700	116407	12.00	0.0012372	0.906143
9800	117754	12.02	0.0012261	0.906260
9900	119095	12.03	0.0012151	0.906325
10000	120455	12.05	0.0012045	0.906511