**Run time analysis of bubble sort and quick sort**

**Student Number:** 1100682

**Student Name:** Geraint Harries

**Year:** 2nd

**Module Code:** CM2303

**Module Name:** Algorithms and Data Structures

**Lecturer Name:** Christine Mumford

**<CONTENTS>**

**Introduction**

This report aims to compare the runtime of two sorting algorithms; bubble sort and quick sort. Both algorithms have different run-time complexities. The complexities I expect for each are as follows:

**Bubble Sort**

Best Case Scenario: *O(n)*

Average Case Scenario: *O(n2)*

Best Case Scenario: *O(n2)*

**Quick Sort**

Best Case Scenario: *O(nlogn)*

Average Case Scenario: *O(nlogn)*

Worse Case Scenario: *O(n2)*

The algorithms will be tested in best case, average case and worse case scenarios

**Design**

**Bubble Sort Pseudocode**

Algorithm bubble–sort3(n, A)

Input: An array, A, of numbers of length n.

Output: The array, A sorted

limit n 1

done 0

while done = 0 do

done 1

for j 1 to limit do

if A[j + 1] < A[j] then

temp A[j]

A[j] A[j + 1]

A[j + 1] temp

done 0

end if

limit limit 1

end for

end while##

**Results**

**Bubble Sort – Best Case**

**A description...**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 10 | 373.92 | 210 | 224.18 | 410 | 359.36 | 610 | 482.28 | 810 | 606.76 |
| 20 | 617.82 | 220 | 230.19 | 420 | 365.9 | 620 | 491.22 | 820 | 614.12 |
| 30 | 1020.31 | 230 | 236.96 | 430 | 373.04 | 630 | 491.43 | 830 | 619.33 |
| 40 | 570.405 \*\* | 240 | 243.02 | 440 | 378.91 | 640 | 498.97 | 840 | 626.82 |
| 50 | 120.5 | 250 | 249.91 | 450 | 385.65 | 650 | 504.97 | 850 | 632.39 |
| 60 | 128.39 | 260 | 256.4 | 460 | 391.47 | 660 | 511.52 | 860 | 639.65 |
| 70 | 133.05 | 270 | 266.8 | 470 | 397.39 | 670 | 518.74 | 870 | 645.47 |
| 80 | 139.44 | 280 | 272.77 | 480 | 403.89 | 680 | 524.4 | 880 | 656.4 |
| 90 | 147.62 | 290 | 279.5 | 490 | 411.64 | 690 | 530.32 | 890 | 658.03 |
| 100 | 153.73 | 300 | 285.94 | 500 | 420.95 | 700 | 537.47 | 900 | 665.71 |
| 110 | 159.84 | 310 | 292.23 | 510 | 417.14 | 710 | 543.32 | 910 | 670.76 |
| 120 | 166.98 | 320 | 358.41 | 520 | 424.05 | 720 | 550.14 | 920 | 678.01 |
| 130 | 172.56 | 330 | 306.45 | 530 | 431.02 | 730 | 555.87 | 930 | 683.2 |
| 140 | 179.45 | 340 | 315.41 | 540 | 436.89 | 740 | 562.92 | 940 | 690.91 |
| 150 | 185.12 | 350 | 320.46 | 550 | 443.93 | 750 | 568.72 | 950 | 696.14 |
| 160 | 192.2 | 360 | 327.3 | 560 | 450.45 | 760 | 576.14 | 960 | 703.39 |
| 170 | 198.68 | 370 | 333.95 | 570 | 455.91 | 770 | 581.59 | 970 | 709.25 |
| 180 | 205.33 | 380 | 339.9 | 580 | 462.25 | 780 | 588.66 | 980 | 716.63 |
| 190 | 211.11 | 390 | 346.97 | 590 | 468.41 | 790 | 594.1 | 990 | 721.56 |
| 200 | 217.6 | 400 | 352.96 | 600 | 475.43 | 800 | 601.36 |  |  |

**Bubble Sort – Average Case**

**A description...**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 10 | 3242.25 | 210 | 70678.9 | 410 | 265693 | 610 | 594899 | 810 | 1067117 |
| 20 | 14552.9 | 220 | 77399.5 | 420 | 279535 | 620 | 618236 | 820 | 1096487 |
| 30 | 27488.6 | 230 | 84210.5 | 430 | 291246 | 630 | 635330 | 830 | 1115295 |
| 40 | 3280.75 | 240 | 91888.6 | 440 | 306150 | 640 | 654278 | 840 | 1149456 |
| 50 | 4920.96 | 250 | 99291.2 | 450 | 319246 | 650 | 680230 | 850 | 1175248 |
| 60 | 6879.78 | 260 | 106224 | 460 | 335896 | 660 | 699531 | 860 | 1206619 |
| 70 | 9036.09 | 270 | 115491 | 470 | 348591 | 670 | 721608 | 870 | 1233758 |
| 80 | 11493.2 | 280 | 124031 | 480 | 367787 | 680 | 744961 | 880 | 1264878 |
| 90 | 14557.0 | 290 | 133103 | 490 | 379661 | 690 | 766445 | 890 | 1293487 |
| 100 | 17609.2 | 300 | 142402 | 500 | 397690 | 700 | 788971 | 900 | 1325376 |
| 110 | 20849.9 | 310 | 152571 | 510 | 412800 | 710 | 812387 | 910 | 1354676 |
| 120 | 24458.5 | 320 | 161599 | 520 | 430122 | 720 | 839002 | 920 | 1384991 |
| 130 | 29492.1 | 330 | 172518 | 530 | 447735 | 730 | 859247 | 930 | 1416586 |
| 140 | 32563.3 | 340 | 182888 | 540 | 464246 | 740 | 885836 | 940 | 1442906 |
| 150 | 37097.1 | 350 | 193244 | 550 | 482022 | 750 | 907730 | 950 | 1476856 |
| 160 | 42066.0 | 360 | 207644 | 560 | 513441 | 760 | 934070 | 960 | 1512899 |
| 170 | 46984.2 | 370 | 215548 | 570 | 527208 | 770 | 960153 | 970 | 1548276 |
| 180 | 52458.4 | 380 | 227799 | 580 | 540693 | 780 | 988462 | 980 | 1569928 |
| 190 | 58292.2 | 390 | 239804 | 590 | 556526 | 790 | 1012333 | 990 | 1609956 |
| 200 | 64586.0 | 400 | 252246 | 600 | 576442 | 800 | 1039766 |  |  |

**Bubble Sort – Worst Case**

**A description...**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 10 | 4940.58 | 210 | 47415.6 | 410 | 173269 | 610 | 375760 | 810 | 652565 |
| 20 | 20867.3 | 220 | 52285.7 | 420 | 181015 | 620 | 387520 | 820 | 668208 |
| 30 | 25090.1 | 230 | 56313.9 | 430 | 189944 | 630 | 399338 | 830 | 684604 |
| 40 | 2487.76 | 240 | 61727.1 | 440 | 198646 | 640 | 410972 | 840 | 699963 |
| 50 | 3518.08 | 250 | 66073.4 | 450 | 207892 | 650 | 424332 | 850 | 717768 |
| 60 | 4970 | 260 | 72015.8 | 460 | 215947 | 660 | 436649 | 860 | 733709 |
| 70 | 6336.93 | 270 | 77850.5 | 470 | 225853 | 670 | 449921 | 870 | 752059 |
| 80 | 8339.5 | 280 | 83171.5 | 480 | 234670 | 680 | 462463 | 880 | 767376 |
| 90 | 9897.16 | 290 | 90477.3 | 490 | 244610 | 690 | 476938 | 890 | 785418 |
| 100 | 12182.6 | 300 | 94830.4 | 500 | 253847 | 700 | 489423 | 900 | 801668 |
| 110 | 14265.6 | 310 | 101228 | 510 | 264361 | 710 | 504068 | 910 | 820262 |
| 120 | 16943.0 | 320 | 107266 | 520 | 273984 | 720 | 517215 | 920 | 838386 |
| 130 | 19321.5 | 330 | 114140 | 530 | 284903 | 730 | 532250 | 930 | 856298 |
| 140 | 22468.2 | 340 | 120643 | 540 | 295001 | 740 | 545872 | 940 | 874262 |
| 150 | 25256.3 | 350 | 127774 | 550 | 306183 | 750 | 561252 | 950 | 893165 |
| 160 | 28770.6 | 360 | 134483 | 560 | 326164 | 760 | 575046 | 960 | 910787 |
| 170 | 31844.4 | 370 | 142136 | 570 | 328369 | 770 | 590874 | 970 | 930341 |
| 180 | 35850.8 | 380 | 149219 | 580 | 338908 | 780 | 605207 | 980 | 948106 |
| 190 | 39281.3 | 390 | 157467 | 590 | 351238 | 790 | 622120 | 990 | 968344 |
| 200 | 45287.1 | 400 | 164804 | 600 | 362126 | 800 | 635959 |  |  |

**Quick Sort – Best Case**

**A description...**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 10 | 1791.5 | 210 | 2960.06 | 410 | 6322.59 | 610 | 9415.44 | 810 | 13185.0 |
| 20 | 3796.07 | 220 | 3003.06 | 420 | 6482.69 | 620 | 9472.2 | 820 | 13141.5 |
| 30 | 5001.43 | 230 | 3091.58 | 430 | 6416.86 | 630 | 9664 | 830 | 12739.0 |
| 40 | 14528.1 | 240 | 3527.81 | 440 | 6339.82 | 640 | 9639.42 | 840 | 13008.0 |
| 50 | 24430.5 | 250 | 3283.5 | 450 | 6559.03 | 650 | 10116.3 | 850 | 13560.3 |
| 60 | 25759.4 | 260 | 3471.81 | 460 | 6791.74 | 660 | 10432.8 | 860 | 13467.3 |
| 70 | 32617.3 | 270 | 3729.5 | 470 | 6809.8 | 670 | 10778.6 | 870 | 13448.4 |
| 80 | 39335.2 | 280 | 3853.46 | 480 | 6847.94 | 680 | 10795.5 | 880 | 13542.5 |
| 90 | 45942.4 | 290 | 4037.31 | 490 | 6966.79 | 690 | 10858.1 | 890 | 13586.2 |
| 100 | 50660.7 | 300 | 4213.71 | 500 | 7124.82 | 700 | 10899.5 | 900 | 13858.1 |
| 110 | 43690.8 | 310 | 4355.13 | 510 | 7145.26 | 710 | 11180.4 | 910 | 14126.1 |
| 120 | 1527 | 320 | 4601.53 | 520 | 7565.85 | 720 | 11529.0 | 920 | 14503.6 |
| 130 | 1605.96 | 330 | 4921.95 | 530 | 7940.01 | 730 | 11687.4 | 930 | 14571.4 |
| 140 | 1839.56 | 340 | 5213.79 | 540 | 8012.11 | 740 | 11770.2 | 940 | 14662.2 |
| 150 | 1917.89 | 350 | 5183.21 | 550 | 8131.79 | 750 | 11793.4 | 950 | 14516.9 |
| 160 | 2123.21 | 360 | 5321.91 | 560 | 8303.62 | 760 | 11882.0 | 960 | 14517.8 |
| 170 | 2377.63 | 370 | 5662.72 | 570 | 8409.78 | 770 | 12030.2 | 970 | 14962.6 |
| 180 | 2545.14 | 380 | 5869.86 | 580 | 8785.53 | 780 | 12485.5 | 980 | 15120.6 |
| 190 | 2651.08 | 390 | 5917.48 | 590 | 9146.75 | 790 | 13088.6 | 990 | 15163.4 |
| 200 | 2849.6 | 400 | 6279.74 | 600 | 9223.85 | 800 | 13163.6 |  |  |

**Quick Sort – Average Case**

**A description...**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 10 | 3463.83 | 210 | 10280.0 | 410 | 20595.2 | 610 | 30738.8 | 810 | 41016.5 |
| 20 | 7952.38 | 220 | 10737.6 | 420 | 21054.3 | 620 | 31341.0 | 820 | 41413.9 |
| 30 | 12212.7 | 230 | 11343.9 | 430 | 21578.6 | 630 | 31890.8 | 830 | 41891.8 |
| 40 | 87293.2 | 240 | 11772.6 | 440 | 22077.2 | 640 | 32249.4 | 840 | 42626.8 |
| 50 | 38003.8 | 250 | 13764.5 | 450 | 22572.1 | 650 | 32858.4 | 850 | 43035.7 |
| 60 | 44380.7 | 260 | 12919.5 | 460 | 23099.9 | 660 | 33431.7 | 860 | 43592.6 |
| 70 | 52184.1 | 270 | 13330.2 | 470 | 23677.1 | 670 | 33798.1 | 870 | 43978.3 |
| 80 | 5804.56 | 280 | 13881.3 | 480 | 24117.7 | 680 | 34342.7 | 880 | 44588.3 |
| 90 | 4103.71 | 290 | 14330.5 | 490 | 24618.0 | 690 | 34902.0 | 890 | 44934.4 |
| 100 | 4648.7 | 300 | 14943.6 | 500 | 25162.4 | 700 | 35343.4 | 900 | 45504.2 |
| 110 | 5122.83 | 310 | 15389.6 | 510 | 25740.0 | 710 | 35771.8 | 910 | 46091.0 |
| 120 | 5635.95 | 320 | 15890.7 | 520 | 26139.0 | 720 | 36413.6 | 920 | 46517.7 |
| 130 | 6116.72 | 330 | 16422.4 | 530 | 26671.6 | 730 | 36937.9 | 930 | 47098.4 |
| 140 | 6640.97 | 340 | 16929.7 | 540 | 27191.8 | 740 | 37366.6 | 940 | 47577.8 |
| 150 | 7592.87 | 350 | 17465.2 | 550 | 27633.3 | 750 | 38022.6 | 950 | 48095.6 |
| 160 | 7654.65 | 360 | 17879.1 | 560 | 28233.8 | 760 | 38468.6 | 960 | 48509.3 |
| 170 | 8190.89 | 370 | 18493.9 | 570 | 28750.4 | 770 | 38882.2 | 970 | 49263.5 |
| 180 | 8735.16 | 380 | 19013.6 | 580 | 29287.2 | 780 | 39367.7 | 980 | 49722.7 |
| 190 | 9231.02 | 390 | 19475.4 | 590 | 29742.2 | 790 | 40012.2 | 990 | 50180.4 |
| 200 | 9707.5 | 400 | 19968.5 | 600 | 30334.6 | 800 | 40555.1 |  |  |

**Quick Sort – Worst Case**

**A description...**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 10 | 2706.86 | 210 | 30864.2 | 410 | 110321 | 610 | 236582 | 810 | 413940 |
| 20 | 12793.4 | 220 | 35112.1 | 420 | 116361 | 620 | 244507 | 820 | 423803 |
| 30 | 28476.6 | 230 | 36651.9 | 430 | 121926 | 630 | 253970 | 830 | 434404 |
| 40 | 40815.1 | 240 | 39781.9 | 440 | 127229 | 640 | 260479 | 840 | 444276 |
| 50 | 14904.4 | 250 | 42976.4 | 450 | 133174 | 650 | 268732 | 850 | 454799 |
| 60 | 3146.98 | 260 | 46179.3 | 460 | 138304 | 660 | 276278 | 860 | 465950 |
| 70 | 4164.43 | 270 | 49644.7 | 470 | 144901 | 670 | 284471 | 870 | 476785 |
| 80 | 5143.41 | 280 | 53363.3 | 480 | 150708 | 680 | 292919 | 880 | 487787 |
| 90 | 6342.11 | 290 | 57004.5 | 490 | 157161 | 690 | 305806 | 890 | 498264 |
| 100 | 7615.58 | 300 | 60820.8 | 500 | 161726 | 700 | 314560 | 900 | 509260 |
| 110 | 9155.83 | 310 | 64872.0 | 510 | 166521 | 710 | 323458 | 910 | 520259 |
| 120 | 10646.0 | 320 | 68862.3 | 520 | 173142 | 720 | 330169 | 920 | 531964 |
| 130 | 12527.0 | 330 | 72779.6 | 530 | 179687 | 730 | 336764 | 930 | 543508 |
| 140 | 14339.8 | 340 | 77407.0 | 540 | 186518 | 740 | 346065 | 940 | 555089 |
| 150 | 16427.5 | 350 | 81721.0 | 550 | 193056 | 750 | 355528 | 950 | 567490 |
| 160 | 18452.6 | 360 | 84487.8 | 560 | 200772 | 760 | 364563 | 960 | 578922 |
| 170 | 20686.5 | 370 | 91163.8 | 570 | 207748 | 770 | 374261 | 970 | 590717 |
| 180 | 22877.2 | 380 | 95779.5 | 580 | 214482 | 780 | 383830 | 980 | 603799 |
| 190 | 25547 | 390 | 100903 | 590 | 221903 | 790 | 393947 | 990 | 615464 |
| 200 | 28483.6 | 400 | 106523 | 600 | 229110 | 800 | 404341 |  |  |

**Testing**

**Bubble Sort**

The code prints out blah blah

Code:

//https://learningcentral.cf.ac.uk/bbcswebdav/pid-2714221-dt-content-rid-2907213\_2/courses/1213-CM2303/SuppNotes2%283%29.pdf

import java.util.Random;

class bubblesort{

public int[] RandArray;

public static Random myRandom = new Random();

public bubblesort (int x) {

RandArray = new int[x];

for(int i = 0; i < x; i++) { //generates random numbers for array size X

RandArray[i] = myRandom.nextInt(100); //Average case

//RandArray[i] = x-i; //worst case

//RandArray[i] = i; //best case

}

}

public void bubble\_srt(int x ){

int t, limit = x - 1;

boolean done = false;

while (done == false){

done = true;

for(int j = 0; j < limit; j++){

if(RandArray[j+1] < RandArray[j]){ //check if the entry in the next array slot is greater than the entry in the current array slot

t = RandArray[j];

RandArray[j]=RandArray[j+1]; //by using temporary memory the two values are switched

RandArray[j+1]=t;

done = false; //makes the system repat the sort if a change was made

System.out.println(RandArray[0] + "," + RandArray[1] + "," + RandArray[2] + "," + RandArray[3] + "," + RandArray[4] + "," + RandArray[5] + "," + RandArray[6] + "," + RandArray[7] + "," + RandArray[8] + "," + RandArray[9]);

}

}

}

}

}

Output:

9,33,69,45,35,32,61,54,79,74

9,33,45,69,35,32,61,54,79,74

9,33,45,35,69,32,61,54,79,74

9,33,45,35,32,69,61,54,79,74

9,33,45,35,32,61,69,54,79,74

9,33,45,35,32,61,54,69,79,74

9,33,45,35,32,61,54,69,74,79

9,33,35,45,32,61,54,69,74,79

9,33,35,32,45,61,54,69,74,79

9,33,35,32,45,54,61,69,74,79

9,33,32,35,45,54,61,69,74,79

9,32,33,35,45,54,61,69,74,79

**Quick Sort**

Blah blah

Code:

//http://www.vogella.com/articles/JavaAlgorithmsQuicksort/article.html

import java.util.Random;

class Quickclock {

public static int[] numbers = {23,67,98,46,17,95,1234,46,11,314,956};

public static void main (String args[]){

quicksort(0, 10);

}

private static void quicksort(int low, int high) {

int i = low, j = high; // Get the pivot element from the middle of the list

int pivot = numbers[low + (high-low)/2]; // bestcase + average case

System.out.println("the pivot is:" + pivot);

if (low >= high) return;

while (i <= j) { // Divide into two lists

// If the current value from the left list is smaller then the pivot

// element then get the next element from the left list

while (numbers[i] < pivot) {

i++;

}

// If the current value from the right list is larger then the pivot

// element then get the next element from the right list

while (numbers[j] > pivot) {

j--;

}

// If we have found a values in the left list which is larger then

// the pivot element and if we have found a value in the right list

// which is smaller then the pivot element then we exchange the

// values.

// As we are done we can increase i and j

if (i <= j) {

exchange(i, j);

i++;

j--;

}

}

System.out.println(numbers[0] + "," + numbers[1] + "," + numbers[2] + "," + numbers[3] + "," + numbers[4] + "," + numbers[5] + "," + numbers[6] + "," + numbers[7] + "," +numbers[8] + "," +numbers[9] + "," + numbers[10]);

// Recursion

if (low < j)

quicksort(low, j);

if (i < high)

quicksort(i, high);

}

private static void exchange(int i, int j) {

int temp = numbers[i];

numbers[i] = numbers[j];

numbers[j] = temp;

}

}

Output:

the pivot is:95

23,67,11,46,17,46,1234,95,98,314,956

the pivot is:11

11,67,23,46,17,46,1234,95,98,314,956

the pivot is:46

11,46,23,17,46,67,1234,95,98,314,956

the pivot is:23

11,17,23,46,46,67,1234,95,98,314,956

the pivot is:46

11,17,23,46,46,67,1234,95,98,314,956

the pivot is:98

11,17,23,46,46,67,98,95,1234,314,956

the pivot is:98

11,17,23,46,46,67,95,98,1234,314,956

the pivot is:314

11,17,23,46,46,67,95,98,314,1234,956

the pivot is:1234

11,17,23,46,46,67,95,98,314,956,1234