CSCI 321 Computer Science III Fall 2018

Assignment 1

1. (60 points) Perform an experimental analysis of the two algorithms prefixAverage1 and prefixAverage2. Visualize their running times as a function of the input size with a chart. Hint: You need to pick several input sizes and run at least 5 tests on each input size to have an accurate estimate on the running time.

import java.util.Scanner;

import java.util.Arrays;

class assign1q1 {

static int cntA = 0;

public static double[] pAvg1(double[] X){

int n = X.length;

double[] a = new double[n];

for(int i = 0; i < n; i++){

cntA++;

double total = 0;

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

cntA++;

total += X[j];

a[i] = total / (i + 1);

}

}

System.out.println("cntA: " + cntA);

return a;

}

static int cntB = 0;

public static double[] pAvg2(double[] X){

int n= X.length;

double [] a = new double[n];

double total = 0;

for (int i = 0; i < n; i ++){

cntB++;

total += X[i];

a[i] = total /(i + 1);

}

System.out.println("cntB: " + cntB);

return a;

}

public static void main(String[] args) {

long startTime = System.currentTimeMillis();

assign1q1 q1= new assign1q1();

Scanner scan1 = new Scanner(System.in);

// int n = scan1;

double[] d= new double[1000];

System.out.println("enter array of size double: ");

int n = scan1.nextInt();

for (int i = 0; i < n; i ++){

d[i] = Math.random();

}

double[] arrA = q1.pAvg1(d);

System.out.println(Arrays.toString(arrA));

double[] arrB = q1.pAvg2(d);

System.out.println(Arrays.toString(arrB));

long endTime = System.currentTimeMillis();

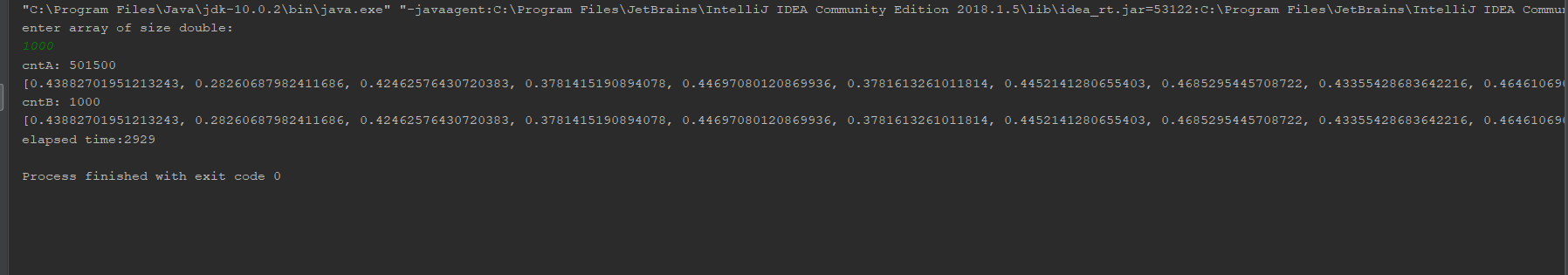
long elapsed = endTime - startTime;

System.out.println("elapsed time:" + elapsed);

}

}

|  |  |  |  |
| --- | --- | --- | --- |
| Array Size: | 10 | 100 | 1000 |
| Run: | Time: | | |
| 1 | 3980 | 3517 | 3176 |
| 2 | 7299 | 3568 | 3275 |
| 3 | 3704 | 4403 | 2964 |
| 4 | 7156 | 2811 | 2355 |
| 5 | 2149 | 2203 | 2929 |



1. (40 points) Textbook (Goodrich 6th edition), Chapter 2 Pg. 100, Problem C-2.24.

//Progression class

public class Progression {

public static void main(String[] args){

Progression prog;

System.out.print("Absolute progression with start values of 2 and 200: ");

prog = new AbsoluteProgression();

prog.printProgression(10);

}

protected long current;

public Progression(long start){

current = start;

}

public long nextValue(){

long answer = current;

advance();

return answer;

}

protected void advance(){

current++;

}

public void printProgression(int x){

System.out.print(nextValue());

for(int i=1 ;i < x; i++){

System.out.print(" " + nextValue());

}

}

}

//AbsProgression class:

public class AbsoluteProgression extends Progression {

protected long prev;

public AbsoluteProgression(){ this(2,200); }

public AbsoluteProgression(long first, long second){

super(first);

prev = first-second;

}

protected void advance(){

long next = Math.abs(current-prev);

prev = current;

current = next;

}

}

