COMP2054-ADE FORMATIVE LAB
Week 3:
"Primitive Operation counting and Graphs"

Objective

Give practice at

- analysing a (non-trivial) program to do counting of primitive operations
- running experiments to look at the scaling behaviour; plot a graph; be able to relate to "Big Oh Theory".

The exercise is similar to last week except that the function is much more interesting, and has a more complicated scaling behaviour.

BEWARE: the primitive operation counting has some subtleties that you need to watch out for !!

Description

Firstly you should download the file Main.java from Moodle.

Your first task should be to read it carefully, and then check that you can compile and run it. Then there are two parts to the lab:

- "**Counting**". In the file "Main.java" you are given a fragment piece of code for a method "p". It contains lines "c += ??" to increment a counter of primitive operations. You are required to modify these increments to do some approximate counting of the "primitive operations" that it performs.

  These should be reported in the table on the next page – along with brief explanation of the increments – especially in those cases that are less obvious.

- "**Graph**". Using the supplied code, modify the Main routine as appropriate, for example changing the range of values of the input size, and the number of runs at each size. Take the resulting outputs and use them to produce an initial scatterplot graph (e.g. in Excel) of how the behaviour changes with n. Then add to the graph extra 'lines' that illustrate the scaling of different aspects of the runtime. The two main aspects of runtime that you should consider are the worst and the best runtime. However, you may also consider the average runtime.

  Examples of "extra lines" are
  1. Lines above the data to show the Big-Oh
  2. Lines below the data to show the Big-Omega

Part A. Table to help keep track of the counting and practice at writing justifications.

Line		Increment:
1	static int p(int[] A) {	<< replace this with your values for the increment; and brief (one line) comments about the ones you think are 'non-obvious'. Do not edit or change the other columns. Do not change font size, or other format . >>
2		
3	int n = A.length; c += 0;	
4	int[] B = new int[n]; c += 0;	
5		
6	int sum = 0; c+= 0;	
7	int max = 0; c += 0 ;	
8		
9	for (int p=0; p < n; p++) { c += 0 ;	
10		
11	int k = A[p]; c += 0 ;	
12	sum += k; c += 0;	
13	B[p] = sum; c += 0;	
14	int m = 0; c += 0 ;	
15	int s; c += 0 ;	
16		
17	s = (n%2 == 0 ? sum : k); c += 0 ;	
18		
19	while (s >= 2) { c += 0 ;	
20	s /= 2; c += 0 ;	
21	m++; c += 0 ;	
22	}	
23		
24	if (m > max) {	
25	max = m; c += 0 ;	
26	}	
27	c += 0;	
28	}	
29	c += 0;	
30		
31	A = B; c += 0;	
32		
33	c += 0; // for the 'return'	
34		
35		
36	return max;}	