Activity 2: Elements of Programming (Part 2)

1. Use and adapt the code PowersOfTwo.java, to print the first 50 powers of 2^N. Include your code as well as the output result.

```
      public static void E1() {

               // read in one command-line argument
           long N = Long.parseLong("50");

           int i = 0;                    // count from 0 to N
           long powerOfTwo = 1;           // the ith power of two

           // repeat until i equals N
           while (i <= N) {
               System.out.println(i + " " + powerOfTwo);    // print out the power
of two
               powerOfTwo = 2 * powerOfTwo;                 // double to get the
next one
               i = i + 1;
           }

      }
```

```
2.  0 1
3.  1 2
4.  2 4
5.  3 8
6.  4 16
7.  5 32
8.  6 64
9.  7 128
10.8 256
11.9 512
12.10 1024
13.11 2048
14.12 4096
15.13 8192
16.14 16384
17.15 32768
18.16 65536
19.17 131072
20.18 262144
21.19 524288
22.20 1048576
23.21 2097152
24.22 4194304
25.23 8388608
26.24 16777216
```

```
27. 25 33554432
28. 26 67108864
29. 27 134217728
30. 28 268435456
31. 29 536870912
32. 30 1073741824
33. 31 2147483648
34. 32 4294967296
35. 33 8589934592
36. 34 17179869184
37. 35 34359738368
38. 36 68719476736
39. 37 137438953472
40. 38 274877906944
41. 39 549755813888
42. 40 1099511627776
43. 41 2199023255552
44. 42 4398046511104
45. 43 8796093022208
46. 44 17592186044416
47. 45 35184372088832
48. 46 70368744177664
49. 47 140737488355328
50. 48 281474976710656
51. 49 562949953421312
52. 50 1125899906842624
```

2. Use and adapt the code PowersOfTwo.java, to print the first 50 powers of 2^N. Include your code as well as the output result.
3. Use the code for RandomWalk.java to create 3 pictures that you like, using the number 100 as argument. To compile, you are required to previously compile StdDraw.java. You will produce 3 plots to be copied into your Activity log document.
4. Use the code Factors.java that prints the prime factors of a number. Follow the examples in the code headings comments and you are required to measure the computation time for the next 6 cases: 3, 6, 9, 12, 15, and 18 digit primes

- java Factors 997
- java Factors 999983
- java Factors 999999937
- java Factors 999999999989
- java Factors 999999999999989
- java Factors 999999999999999989

You are free to modify the source code to include a timing function. Here is an example you can review at stackoverflow.com. Include source code and output in your working document.

4. Use the program FunctionGrowth.java that prints a table of the values of *log N*, *N*, *N log N*, N^2, N^3, and 2^N for *N* = 16, 32, 64, ..., 2048. What are the limits of this code? Suppose we want to stop not at N=2048. but at N=1073741824. Modify your code to do this. Add the modified code to your document and include generated output.
5. Modify the code Binary.java that converts any number to binary form, to convert any number to its hexadecimal form. Print the first 256 numbers in hex. Include code and output in your working document.
6. Modify the code DayOfWeek.java to print the Day of the Week (Sunday, Monday, ...).
7. Let's play cards. Use the code Deal.java to play 21 or BlackJack for 2 users. You are always the first deal of cards, the house the second. Modify the code to ask for an additional card (Hit=1) or none (Stay=0) for the user. In 20 trials, how many times did you beat the house?. Add the modified code to your working document and describe your experience.
8. Use the code Birthday.java, to run at least 20 experiments and compute the average number of people needed to show up in a room in order that 2 people share the same birthday.
9. Use the code to build the Pascal triangle, Pascal.java. Produce a Pascal Triangle to level 10

```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1
1 7 21 35 35 21 7 1
1 8 28 56 70 56 28 8 1
1 9 36 84 126 126 84 36 9 1
```

10. You are required to run the code that generates a Sierpinski triangle: Sierpinski.java. This code requires compiling beforehand DrawingPanel.java. Can you guess an algorithm that counts how many solid black inverted triangles and how many upright white triangles per level N. Justify your answer.

```
System.out.println("El numero de triangulos negros es: " + Math.pow(3 ,
level));
```