3. Do not make use of any variables in your answers to this question.
Again, only consider positive numbers.

a. Write a function `parta(int x)` which prints a sequence of numbers starting with `x` itself, and counting down, but stopping after it first prints a number that is divisible by 11.

For example, `parta(30)` should print `30 29 28 27 26 25 24 23 22`

b. Write another function `partb(int x)` which does not print anything, but instead returns as its result the length of the sequence that `parta(x)` would have printed.

For example, `const int n = partb(30)` should set `n` to 9, because `parta(30)` prints 9 numbers.

c. There is a mysterious mathematical thing called a *Collatz Sequence*. It can start with any number N. If N is odd, the rest of the sequence is the collatz sequence starting from 3×N+1, but if N is even, the rest of the sequence is the collatz sequence starting from N/2. The sequence stops after it reaches 1.

Write a function `partc(int N)` which prints the entire Collatz sequence starting from N.

For example, `partc(24)` should print `24 12 6 3 10 5 16 8 4 2 1`

d. Write another function `partd(int N)` which returns as its result the length of the Collatz sequence that starts with N.

For example, `partd(24)` is equal to 11, because as you can see from the previous example, the Collatz sequence starting from 24 consists of 11 numbers.

3. Do not make use of any variables in your answers to this question.
Use recursive functions to cause repetition when needed.

For this parts a to c you will be writing some functions that process arrays of positive ints. The arrays can be of any length, but will always have a value of `-1` added as a sentinel to indicate the end of the data. The `-1` is not part of the data to be processed, it just marks the end.

Here are two example arrays that your function might have to work on:

```
const int X[] = { 9, 5, 3, 7, 2, 4, 3, 4, -1 };
const int Y[] = { 1, 0, 2, 0, -1 };
```

You may always make use of previous answers

You may always give your functions extra parameters *if needed*.

a. Write a function that can take any such array as its parameter, and print all the data it contains (not including the `-1`) on a single line.

For example, `answera(X)` should print `9 5 3 7 2 4 3 4`
`answera(Y)` should print `1 0 2 0`

b. Write a function that given such an array as its parameter, returns as its result the number of data items in the array.

For example, the value of `answerb(X)` should be 8
the value of `answerb(Y)` should be 4

c. Write a function that given such an array as its parameter, calculates and prints the average of all the data items in the array

For example, `answerc(X)` should print `4.675`
`answerc(Y)` should print `0.75`

d. Write a function that takes a `double` A and an `int` B as its parameters. It should calculate and return as its result the value of A to the power of B. Do not use logarithms.

For example, the value of `answerd(3.0, 4)` should be 81.0