1. **Input**: An array $A[1 \dots n]$ only containing the values 0 and 1
   Output: *True* if there are more 1s than 0; otherwise *False*

2. **Input**: Array $A[1 \dots n]$ of numbers
   Output: Sum of even numbers in A

3. **Input**: Array $A[1 \dots n]$ of numbers
   Output: Sum of *even-indexed* numbers in A

4. Write an algorithm in pseudo-code for multiplication *a la russe*.

5. **Input**: An array $A[1, \dots, n]$
   Output: The index of the second-largest element in A

   Question: can you adapt your strategy to give back the third-biggest element of A? What about the fourth-biggest?

6. **Input**: An array $A[1, \dots, n]$ with n odd and all $A[i]$ *distinct*: if $i \neq j$ then $A[i] \neq A[j]$
   Output: The middle value of the array A

   Let's begin by understanding the problem. If the array is $[1, 2, 0]$ we should return 1. If it's $[3, 7, 9, 5, 2]$ we should return 5.

   Idea 1: the middle value is basically the $(\frac{n}{2}+1)$th[1] value, and we know how to find the first, second, and third value, so maybe we can use the idea above. But then we need to remember quite a lot of values, so we'll definitely need another array. Which means we have to change our specification:

   Input: An array $A[1, \dots, n]$ with n odd and all $A[i]$ *distinct*: if $i \neq j$ then $A[i] \neq A[j]$, and an array $B[1, \dots, m]$ with $m = \frac{n}{2} + 1$
   Output: The middle value of the array A

7. **Input**: An array $A[1, \dots, n]$ with $n \geq 1$
   Output: The number that occurs most times in A

[1] remember that when we divide we ignore the fraction, so we always round down