Recursion

1. Given a positive number. Write a recursive function that returns its factorial. ( n! = n \* (n-1) \* (n-2) \* … \* 3 \* 2 \* 1 , 0! = 1 )

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5 | “5! = 120” |
| 1 | “1! = 1” |
| 7 | “7! = 5040” |

1. Given an integer. Write a recursive function that returns **n**-th element in the Fibonacci sequence.

|  |  |
| --- | --- |
| **Input** | **Output** |
| 0 | 0 |
| 1 | 1 |
| 2 | 1 |
| 3 | 2 |
| 21 | 10946 |

1. Given an array. Write a recursive function that returns the sum of it. (Hint: *arr.pop()*)

|  |  |
| --- | --- |
| **Input** | **Output** |
| [] | 0 |
| [1, -1, 35] | 35 |
| [1, 10, 12, 3] | 26 |

1. Given a number. Write a recursive function that reverse the number. Return the new number.

|  |  |
| --- | --- |
| **Input** | **Output** |
| 2 | 2 |
| 13 | 31 |
| 815796 | 697518 |

1. Given a number and an array. Find the second occurrence of the number in the array. Consider that the occurrence of each element in the array is at least two.

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5, [0, -1, 0, 5, 6, 6, 5, -1, 0, 5, 6] | 6 |
| 8, [8, 8, 4, 0, 8, 0, 0, 0, 4] | 1 |

1. Given a substring and a string. Find how many times the substring occurred in the string. (For getting substring of the string use *str.substring(startIndex, endIndex), str.substr(startIndex, length)* )

|  |  |
| --- | --- |
| **Input** | **Output** |
| ‘ar’, ‘Are var far shared?’ | 3 |
| ‘ook’, ‘Facebook looks odd’ | 2 |

1. Given a string, compute recursively (no loops) a new string where all appearances of "pi" have been replaced by "3.14".

|  |  |
| --- | --- |
| **Input** | **Output** |
| “Picsart pipelines” | “3.14csart 3.14pelines” |
| “picturespicturespictures” | “3.14ctures3.14ctures3.14ctures” |

1. Given a string. Return true, if it is mirror symmetric to its middle symbol.

|  |  |
| --- | --- |
| **Input** | **Output** |
| “abcddcba” | true |
| “abcdabcd” | false |

1. Given an array. Write a recursive function that removes the first element and returns the given array. (without using *arr.unshift()*)

|  |  |
| --- | --- |
| **Input** | **Output** |
| [6, 78, ‘n’, 0, 1] | [78, ‘n’, 0, 1] |
| [5] | [] |
| [] | [] |

1. Given an array of nested arrays. Write a recursive function that flattens it. (Hint create function that concats arrays).

|  |  |
| --- | --- |
| **Input** | **Output** |
| [1, [3, 4, [1, 2]], 10] | [1, 3, 4, 1, 2, 10] |
| [14, [1, [[[3, []]], 1], 0] | [14, 1, 3, 1, 0] |

1. Given an array and a number N. Write a recursive function that rotates an array N places to the left. (*Hint*: to remove element from the beginning use *arr.unshift()*)

|  |  |
| --- | --- |
| [‘a’, ‘b’, ‘c’, ‘d’, ‘e’, ‘f’, ‘g’, ‘h’] 3 | [‘d’, ‘e’, ‘f’, ‘g’, ‘h’, ‘a’, ‘b’, ‘c’] |
| [‘a’, ‘b’, ‘c’, ‘d’, ‘e’, ‘f’, ‘g’, ‘h’] -2 | [‘g’, ‘h’, ‘a’, ‘b’, ‘c’, ‘d’, ‘e’, ‘f’] |

1. Given a number. Write a function that calculates its sum of the digits and if that sum has more than 1 digit find the sum of digits of that number. Repeat that process if needed and return the result.

|  |  |
| --- | --- |
| **Input** | **Output** |
| 14 | 5 |
| 29 | 2 |
| 999999999999 | 9 |