Problem Domain

- Write a function called reverseArray which takes an array as an argument. Without utilizing any of the built-in methods available to your language, return an array with elements in reversed order.

Input List of Numbers

Output Reversed List of Numbers

Test Cases

1. Given [1, 2, 3, 4, 5, 6] Return [6, 5, 4, 3, 2, 1]

2. Given [89, 2354, 3546, 23, 10, -923, 823, -12] Return [-12, 823, -923, 10, 23, 3546, 2354, 89]

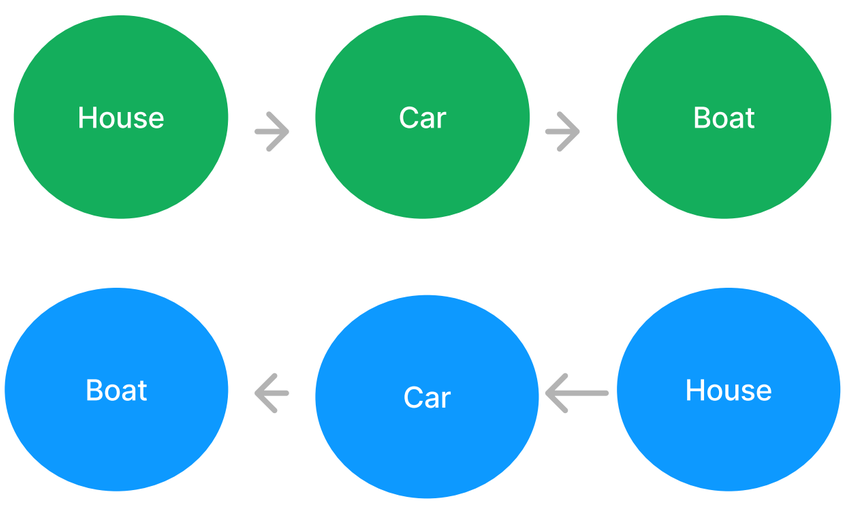
3. Given [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199]

   Return [199, 197, 193, 191, 181, 179, 173, 167, 163, 157, 151, 149, 139, 137, 131, 127, 113, 109, 107, 103, 101, 97, 89, 83, 79, 73, 71, 67, 61, 59, 53, 47, 43, 41, 37, 31, 29, 23, 19, 17, 13, 11, 7, 5, 3, 2]

4. Given ['House', 'Car', 'Boat'] Return ['Boat, 'Car', 'House']

5. Given 3 Return Null

Visualization



Algorithm

0. Check for list, if no list return Null
1. Take in a List
2. Append every index in List to new List in reverse order
3. Return new List

Big O

Time → 0(n) Because this is a 1 to 1 relationship, that growing proportional to the original List

Space → 0(2n) for every one item added to the list it will take 1 action to complete

Pseudo

```
  def reverse_list(list): #
    if type(list) != list: #
    return "not a list"
      new_list = []
      for x in range(len(list)):

      new_list.append(list[-(x+1)])
      print(new_list)
          return new_list

      sample = ["car", "boat", "house"]
       print(reverse_list(sample))
```

Step Through

List	['house','car','boat']	[1,2,3,4,5]	Car
new_list	['boat','car','house']	[5,4,3,2,1]	Null

x	0	1	2	3	4
-(x+1)	-1	-2	-3	-4	-5
list[-(x+1)]	boat	car	house		
new_list	[boat]	[boat, car]	[boat, car, house]		