Reverse a stack using recursion

Write a program to reverse a stack using recursion. You are not allowed to use loop constructs like while, for..etc, and you can only use the following ADT functions on Stack S:
isEmpty(S)
push(S)
pop(S)

Recommended: Please try your approach on ***{IDE}*** first, before moving on to the solution.

The idea of the solution is to hold all values in Function Call Stack until the stack becomes empty. When the stack becomes empty, insert all held items one by one at the bottom of the stack.
For example, let the input stack be

```
1  <-- top
2
3
4
```

First 4 is inserted at the bottom.
4 <-- top

Then 3 is inserted at the bottom
4 <-- top
3

Then 2 is inserted at the bottom
4 <-- top
3
2

Then 1 is inserted at the bottom
4 <-- top
3
2
1

So we need a function that inserts at the bottom of a stack using the above given basic stack function.
void insertAtBottom((): First pops all stack items and stores the popped item in function call stack using recursion. And when stack becomes empty, pushes new item and all items stored in call stack.
void reverse(): This function mainly uses insertAtBottom() to pop all items one by one and insert the popped items at the bottom.

```
def reverse(arr):
    if len(arr):
        temp = arr.pop()
        reverse(arr)
        insertAtBottom(arr,temp)


def insertAtBottom(arr,item):
    if not len(arr):
        arr.append(item)
    else:
        temp = arr.pop()
        insertAtBottom(arr,item)
        arr.append(temp)


arr = [1,2,3,4,5,6,7,8,9]
reverse(arr)
print(arr)
```