Van Robbins
CSCI-C 310
Dr. Hong Liu
5/21/2025

Module 1: Recursion Assignment

Task 1: Nth Fibonacci Number

Description: Write a recursive function that calculates the nth Fibonacci number, where n is a positive integer. The Fibonacci sequence is a series of numbers in which each number is the sum of the two preceding ones, starting with 0 and 1.

Solution:

```python
def nth_fib(num):
    # Base case: the first two Fibonacci numbers are 0 and 1
    if num <= 2:
        return num - 1
    # Recursive case: sum of the two preceding Fibonacci numbers
    return nth_fib(num - 1) + nth_fib(num - 2)
```

Test Cases:

```python
print(nth_fib(10)) #Output: 34

print(nth_fib(5)) #Output: 3

print(nth_fib(3)) #Output: 1
```

Output:

```
PS C:\Users\Van\Documents\School\Summer2025\C310> &
34
3
1
```

Van Robbins
CSCI-C 310
Dr. Hong Liu
5/21/2025

Module 1: Recursion Assignment

Task 2: Reverse String

Description: Write a recursive function that reverses a string. The function should take a string as input and return the reversed string as output.

Solution:

```python
def reverse(high, low, string_li):
    # Convert input to list if it's a string (for mutability)
    if type(string_li) == str:
        string_li = list(string_li)
    # Base case: when the pointers cross or meet, the string is fully reversed
    if high <= low:
        return ''.join(string_li)
    else:
        # Swap the characters at positions high and low
        temp = string_li[high]
        string_li[high] = string_li[low]
        string_li[low] = temp
        # Recursive call, moving the pointers towards the center
        return reverse(high - 1, low + 1, string_li)
```

Test Cases:

```python
a = "Hello World"
print(reverse(len(a) - 1, 0, a))  #Output: dlroW olleH

b = "Indiana University"
print(reverse(len(b) - 1, 0, b)) #Output: ytisrevinU anaidnI

c = "Racecar"
print(reverse(len(c) - 1, 0, c)) #Output: racecaR
```

Output:

```
PS C:\Users\Van\Documents\School\Summer2025\
dlroW olleH
ytisrevinU anaidnI
racecaR
```

Van Robbins
CSCI-C 310
Dr. Hong Liu
5/21/2025

Module 1: Recursion Assignment

Task 3: Binary Search

Description: Write a recursive function binary_search(arr, target) that performs a binary search on a sorted array arr to find the index of the given target value. If the target is found, return its index; otherwise, return -1.

Solution:

```python
def binary_search(arr, target, offset=0):
    # Base case: if the array is empty, target is not found
    if(len(arr)==0):
        return -1;
    # Find the middle index int
    mid_Idx=len(arr)//2
    # If the middle element is the target, return its index (adjusted by offset)
    if(arr[mid_Idx]==target):
        return offset+mid_Idx;
    # If the middle element is less than the target, search the right half
    elif(arr[mid_Idx]<target):
        # Increase offset since we're skipping the left half including mid_Idx
        return binary_search(arr[mid_Idx+1:],target,offset+mid_Idx+1)
    else:
        # Otherwise, search the left half (offset remains the same)
        return binary_search(arr[:mid_Idx],target, offset)
```

Test Cases:

```python
array_a = [1, 3, 5, 7, 9, 11, 13, 15]
target_a = 7
print(binary_search(array_a,target_a))  # Output: 3

array_b = [2, 4, 6, 8, 10]
target_b = 5
print(binary_search(array_b,target_b))  # Output: -1

array_c = [4, 12, 18, 27, 34, 56, 67, 89]
target_c = 56
print(binary_search(array_c, target_c)) # Output: 5
```

Output:

```
PS C:\Users\Van\Documents\School\Summer2025\C310>
3
-1
5
```