ชื่อ – นามสกุล .....เอกรินทร์ องอาจ.....รหัสนักศึกษา.....64015172.....

Data Structures and Algorithm

ภาควิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

การทดลองที่ 9 : Recursive

จุดประสงค์

1. นักศึกษาเข้าใจการทำงานของ Recursive Operation และสามารถออกแบบฟังก์ชันที่ทำงานเป็นแบบ Recursive ตามต้องการได้

ตอนที่ 1 : การออกแบบการทำงานและสร้างฟังก์ชันที่ทำงานแบบ Recursive

1. ให้นักศึกษาเขียนฟังก์ชัน

def isPalindrome(str):

โดย str เป็น string ที่ต้องการตรวจสอบและส่งค่าเป็น true เมื่อ str เป็น palindrome และให้ค่า false เมื่อ str ไม่เป็น palindrome

แนวคิดในการออกแบบการทำงานแบบ recursive

เปรียบเทียบ Array ตัวหน้า ตัวท้าย ถ้าเหมือนกัน POP ออก return true ไม่ตรงเงื่อนไข เรียกซ้ำจนกว่าจะ pop ออกจนเหลือตัวเดียว

กรณีของ str ที่เป็น base case คืออะไร และ ที่ base case จะมีการทำงานอย่างไร

str len == 1

ชื่อ – นามสกุล .....เอกรินทร์ องอาจ.....รหัสนักศึกษา.....64015172.....

กรณีของ str ที่เป็น recursive case จะมีการทำงานอย่างไร

เรียกซ้ำ ถอดออกไปเลื่อย ๆ จนกว่าจะเหลือ 1 ตัว

ฟังก์ชันที่ได้

ทดสอบการทำงานของฟังก์ชัน

str	**ผลลัพธ์**
abcdcba	True
atoyota	True
kmitl	false
manassanan	false
programming	false
fundamental	false

ชื่อ – นามสกุล .....เอกรินทร์ องอาจ.....รหัสนักศึกษา.....64015172.....

2. ให้นักศึกษาเขียนฟังก์ชัน

def isAscending(list_of_integer):

โดย list_of_integer เป็น list ของข้อมูลตัวเลขจำนวนเต็มที่ต้องการตรวจสอบว่าเป็น list ที่เรียงลำดับจากน้อยไปมากแล้วหรือไม่ และส่งค่าเป็น true เมื่อข้อมูลใน list_of_integer เรียงลำดับจากน้อยไปมาก และให้ค่า false เมื่อ list_of_integer ไม่ได้เรียงลำดับจากน้อยไปมาก

แนวคิดในการออกแบบการทำงานแบบ recursive

เปรียบเทียบ ถ้าตัว I < 2 แล้วค่อย ๆ กระเทิบไปเรื่อย ๆ จนกว่าจะ n + 1 จะเท่ากับ len(list) ถ้าเกิดตัว 1 !< 2 จะ return false n ก็จะไม่เท่ากับ len(list)

กรณีของ list ที่เป็น base case คืออะไร และ ที่ base case จะมีการทำงานอย่างไร

n == len(list)

กรณีของ list ที่เป็น recursive case จะมีการทำงานอย่างไร

จะคล้ายเดิม ถ้ามีการ update ค่า อันที่เรียกครั้งต่อไปก็จะรู้ว่ามีการ update ค่าตามไปด้วย

ชื่อ – นามสกุล .....เอกรินทร์ องอาจ.....รหัสนักศึกษา.....64015172.....

ฟังก์ชันที่ได้

```python
def isAscending(list_of_integer, n):
    if n == len(list_of_integer)-1:
        return True
    else:
        if list_of_integer[n] <= list_of_integer[n+1]:
            return isAscending(list_of_integer, n+1)
        else:
            return False


print(isAscending([1, 2, 3, 4, 5, 6, 7], 0))
print(isAscending([3, 4, 2, 5, 6, 1, 2], 0))
print(isAscending([9, 8, 7, 6, 5, 4], 0))
print(isAscending([0, 0, 1, 1, 2, 2, 3, 3, 4, 4, 5, 5], 0))
print(isAscending([6, 7, 8, 9, 10, 11, 12], 0))
print(isAscending([6, 3, 8, 7, 9, 2, 3, 1, 5], 0))
```

Snipped

ทดสอบการทำงานของฟังก์ชัน

list_of_integer	ผลลัพธ์
[1,2,3,4,5,6,7]	True
[3,4,2,5,6,1,2]	False
[9,8,7,6,5,4]	False
[0,0,1,1,2,2,3,3,4,4,5,5]	True
[6,7,8,9,10,11,12]	True
[6,3,8,7,9,2,3,1,5]	False

3. ให้นักศึกษาเขียนฟังก์ชัน

def group_of_no_1(island_list,point_no):

โดย

island_list คือ list ของตัวเลข 1 และ 0 ที่เรียงตัวกัน

point_no คือ ตำแหน่งที่ต้องการตรวจสอบ

ฟังก์ชัน group_of_no_1 จะหาจำนวนตัวเลข 1 ที่ตำแหน่ง point_no ว่ามีจำนวนเลข 1 ติดกันกี่ตัว

ถ้า island_list = [0,1,1,1,1,0,0,1,1,0,1,0] และ point_no = 2 จะได้ผลลัพธ์คือ 4 เพราะมีตัวเลข 1 เรียงกัน 4 ตัวที่ตำแหน่งนั้น หรือถ้า point_no = 8 จะได้ผลลัพธ์เป็น 2

แนวคิดในการออกแบบการทำงานแบบ recursive

เช็คตัวเริ่มต้น ถ้า n-1 เหลือ n+1 เหมือนตัวกลางของมันจะไม่กระเทิบ แล้วจะทำ mark ตัวที่เช็คไปแล้ว ถ้าตรง จะเปลี่ยนตัวนั้นเป็น 2 กันการเรียกซ้ำ จนแรมเต็ม ถ้าเจอก็จะส่งค่า +1 กลับมาเรื่อย ๆ

กรณีของ base case คืออะไร และ ที่ base case จะมีการทำงานอย่างไร

Base case คือ array

ชื่อ – นามสกุล .....เอกรินทร์ องอาจ.....รหัสนักศึกษา.....64015172.....

กรณีของ recursive case จะมีการทำงานอย่างไร

ถ้าเจอเลข 1 จะทำการเปลี่ยนเลข 1 เป็น 2 ก่อนแล้วค่อย recursive

ฟังก์ชันที่ได้

```py
def group_of_no_1(island_list, point_no):
    if point_no < 0 or point_no == len(island_list):
        return 0
    if island_list[point_no] == 1:
        island_list[point_no] = 0
        return 1 + group_of_no_1(island_list, point_no-1) + group_of_no_1(island_list, point_no+1)
    else:
        return 0

print(group_of_no_1([1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0], 1))
print(group_of_no_1([1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0], 5))
print(group_of_no_1([1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1], 4))
print(group_of_no_1([1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1], 10))
print(group_of_no_1([1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1], 1))
print(group_of_no_1([0, 1, 0, 1, 0, 1, 0, 1, 0, 1], 7))
```

Snipped

ทดสอบการทำงานของฟังก์ชัน

island_list	point_no	ผลลัพธ์
[1,1,1,1,0,0,0,1,1,1,0,0]	1	4
[1,1,1,1,0,0,0,1,1,1,0,0]	5	0
[1,0,1,1,1,0,0,0,1,1,1,1,1,1]	4	3
[1,0,1,1,1,0,0,0,1,1,1,1,1,1]	10	6
[1,0,1,1,1,0,0,0,1,1,1,1,1,1]	1	0
[0,1,0,1,0,1,0,1,0,1]	7	1

4. ให้นักศึกษาเขียนฟังก์ชัน

def valid_parentheses(str):

โดย

str คือ string ของวงเล็บในรูปแบบต่างๆ

ฟังก์ชั่น valid_parentheses จะตรวจสอบว่าวงเล็บใน str มีการจัดลำดับได้อย่างถูกต้องหรือไม่

ถ้า str = “(()()(()))”จะได้ผลลัพธ์คือ True , ถ้า str = “(((()()(()))”จะได้ผลลัพธ์คือ False

แนวคิดในการออกแบบการทำงานแบบ recursive

ถ้าเจอ () จะแทนค่าเป็น ช่องว่าง แล้ว recursive แล้ววนแทนช่องว่างไปเรื่อยจนไม่มี หลังจากนั้นจะ return true

กรณีของ base case คืออะไร และ ที่ base case จะมีการทำงานอย่างไร

จนกว่า Str จะเท่ากับ ช่องว่าง

ชื่อ – นามสกุล .....เอกรินทร์ องอาจ.....รหัสนักศึกษา.....64015172.....

กรณีของ recursive case จะมีการทำงานอย่างไร

ถ้าเจอวงเล็บจะแทนตำแหน่งนั้นด้วยช่องว่างก่อนจะ recursive

ฟังก์ชันที่ได้

ทดสอบการทำงานของฟังก์ชัน

str	ผลลัพธ์
(()()(()())	True
((()()	False
())()()(	False
(((()))(((())))	True
()()(((())))	True
()	True