حل مسئله

دوره آموزشی پایتون

جلسه هشتم

عبداله صارمی نایینی

توابع بازگشتی

تعریف

به تابعی بازگشتی می گوییم که در روند اجرا خودش را فراخوانی کند . شاید به نظر برسد که در چنین حالتی برنامه هیچوقت به پایان نمیرسد ولی همیشه اینطور نیست اگر تابع ما شرایط خاصی داشته باشد و حالت پایه برای آن تعریف شده باشد روند اجرای آن به پایان میرسد و نتیجه مورد نظر بدست می آید.

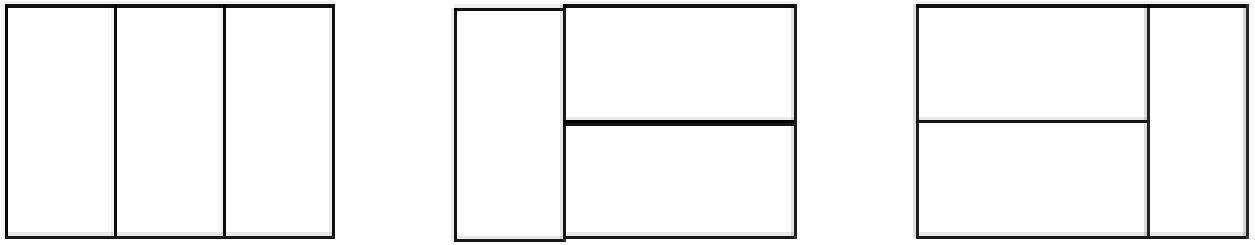
- باید برای مسئله حداقل یک ***حالت پایه*** تعریف کرده و مقدار تابع را در آن حالت بدانیم.
- روند فراخوانی ها باید به حالت(های) پایه ختم شود و به دور منجر نشود.

تابع فاکتوریل

```
def fact(n):
    if n==0:
        return 1
    else:
        return n*fact(n-1)
```

fact(5)=5*fact(4)=
5*4*fact(3)=
5*4*3*fact(2)=
5*4*3*2*fact(1)=
5*4*3*2*1*fact(0)=
5*4*3*2*1*1
=120

۷. تعداد روش‌های پر کردن یک مستطیل $۲\times n$ با n دومینو با ابعاد ۲×۱ را j_n می‌نامیم. این تأکید که هیچ خانه‌ای نباید اضافه یا خالی بماند برای مثال $n = ۳ ، j = ۲$



الف) برنامه‌ای برای محاسبه‌ی j_n بنویسید.

ب) تابع بازگشتی برای محاسبه‌ی آن بنویسید.

ج) سعی کنید برنامه‌ای غیر بازگشتی برای این کار پیدا کنید؛

J(n-1)

J(n)

J(n-1)

j(n-2)

$$J(n)=J(n-1)+J(n-2)\ ,\ J(1)=1\ ,\ J(2)=2$$

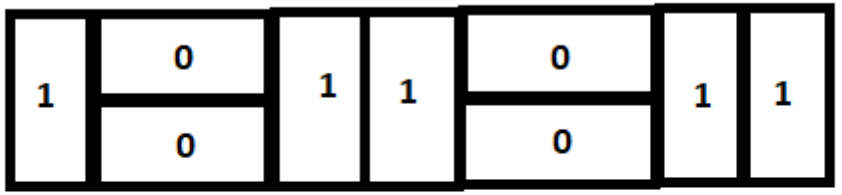
الف

```
def J(n):
    j1=1
    j2=2
    for i in range(2,n):
        temp=j1+j2
        j1=j2
        j2=temp
    return j2
```

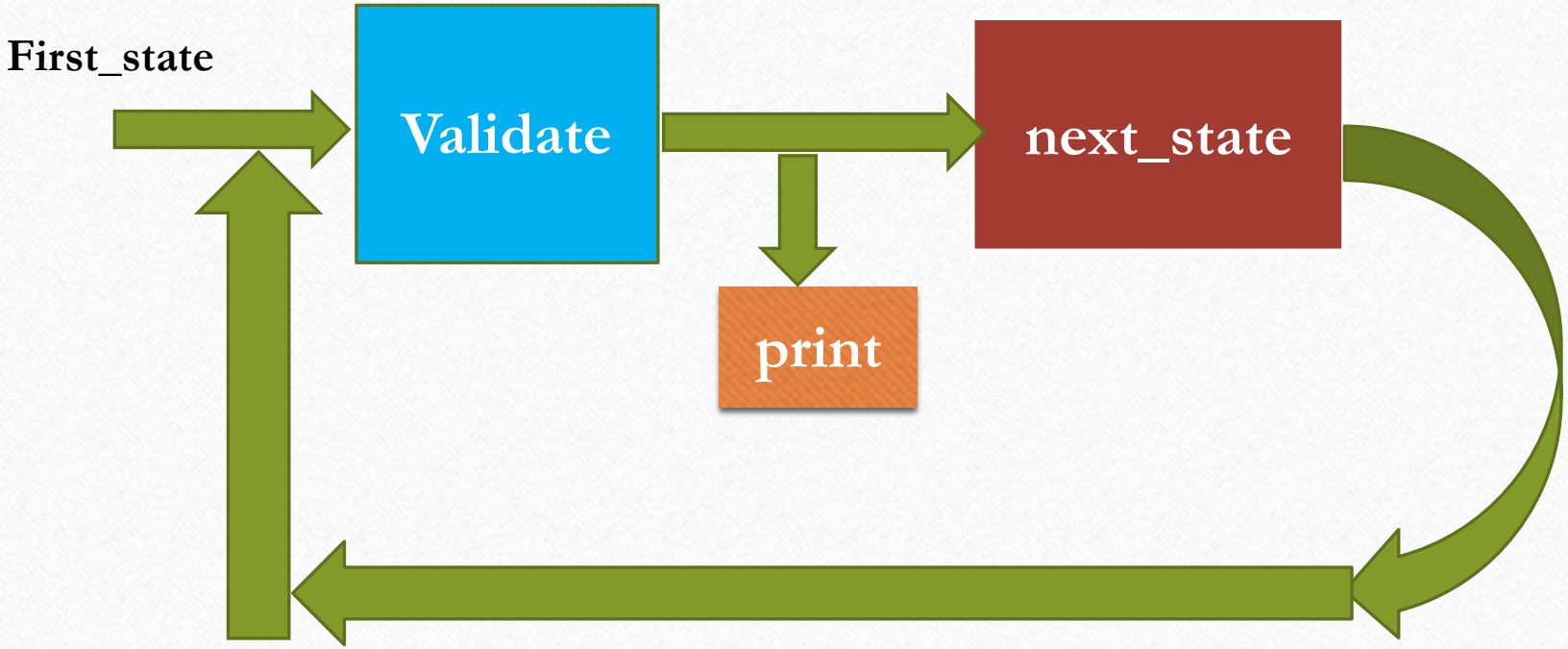
ب

```
def J(n):
    if n<=2:
        return n
    else:
        return J(n-2)+J(n-1)
```

ج



state= 100110011


First_state
Validate
next_state
print

```
def first_state(n):
    return [0 for i in range(n)]
```

```
def next_state(current_state):
    for i in range(len(current_state)-1,-1,-1):
        if current_state[i]==0:
            current_state[i]=1
            return current_state
        else:
            current_state[i]=0
    return None
```

```
def validate(current_state):
    count=0
    even=0
    val=True
    for i in range(len(current_state)):
        if current_state[i]==0:
            count=count+1
    if count % 2==1:
        val=False
    else:
        i=0
        while i<len(current_state)-1:
            if current_state[i]==0 and current_state[i+1]==0:
                even=even+1
                i=i+1
            i=i+1
        if even*2!=count:
            val=False
    return val
```

```
state=first_state(6)
while state!=None:
    if validate(state):
        print(state)
    state=next_state(state)
```

```
[0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 1, 1]
[0, 0, 1, 0, 0, 1]
[0, 0, 1, 1, 0, 0]
[0, 0, 1, 1, 1, 1]
[1, 0, 0, 0, 0, 1]
[1, 0, 0, 1, 0, 0]
[1, 0, 0, 1, 1, 1]
[1, 1, 0, 0, 0, 0]
[1, 1, 0, 0, 1, 1]
[1, 1, 1, 0, 0, 1]
[1, 1, 1, 1, 0, 0]
[1, 1, 1, 1, 1, 1]
```

پایان

از توجه شما متشکرم