6.5 递归

函数调用自身的编程技巧称为递归

递归做为一种算法在程序设计中广泛应用。它通常把一个大型复杂的问题层层转化为一个与原问题相似的规模较小的问题来求解，大大减少了程序的代码量。

递归的能力在于用有限的语句来定义对象的无限集合。一般来说，递归需要终止条件和递归条件。当终止条件不满足时，递归前进；当终止条件满足时，递归返回。编写递归函数时，必须告诉它何时停止递归，从而避免形成无限循环。

斐波那契数列递归定义

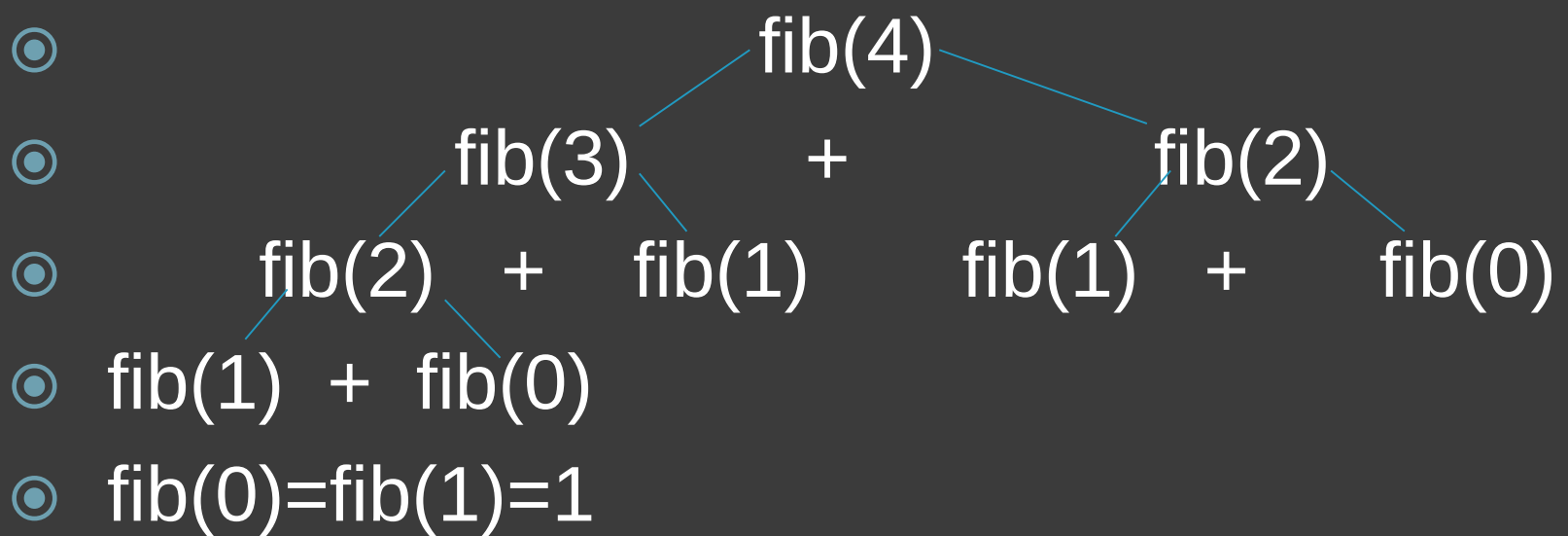
- 斐波那契数列是一个经常使用递归方式定义的序列.
- fib(0)= 1　　n=0　　(1)
- fib(1)= 1　　n=1　　(2)
- fib(n)=fib(n-1)+fib(n-2)　n>=2　　(3)
- 公式（1）和公式（2）是递归终止条件，公式（3）是递归式

斐波那契数列递归程序

```
def  fib(n): #假定n是正整数,返回第n+1个斐波那契数
    if n == 0 or n == 1:
        return 1
    else:
        return fib(n-1) + fib(n-2)
print(fib(4))
```

- 实验：fib(100)要算多久？

斐波那契递归函数调用过程



- fib(0)=fib(1)=1

程序运行

```
def  fib(n): #假定n是正整数,返回第n+1个斐波那契数
    if n == 0 or n == 1:
        return 1
    else:
        return fib(n-1) + fib(n-2)
print(fib(4))
```

改进斐波那契数列递归程序

```
pre={0:1,1:1}
def fib(n):
    if n in pre:  #可以用in检查字典中是否有n这个关键字
        return pre[n]
    else:
        newvalue=fib(n-1)+fib(n-2)
        pre[n]=newvalue #增加字典的条目
        return newvalue
print(fib(4))
print(fib(100))
```

输出结果：

5

573147844013817084101

求嵌套列表中数字元素和(返回多个值）

```
def flatten(items):
    lst=[]
    for x in items:
        if isinstance(x,(list,tuple)) and not isinstance(x,str):
            for element in flatten(x):
                lst.append(element)
        else:
            if type(x)!=str:
                lst.append(x)
    return lst

items=[11,2,[3,7],(68,-1),"123",9]
l=[i for i in flatten(items)]
print(l)
print(sum(l))
```

程序输出：

```
[11, 2, 3, 7, 68, -1, 9]
99
```