

## Tracing a Program with Boxes and Arrows

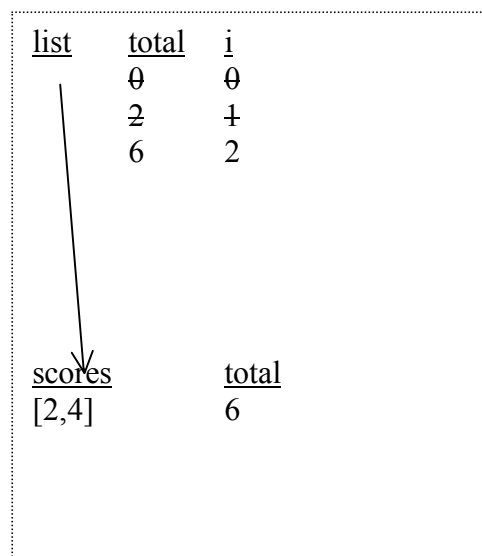
### Guidelines

1. Start execution in the 'main' program.
2. When a new variable is referred to create a memory cell (name of variable with a line under it) for it and place its value in it. Put the memory cell to the right of the main code.
3. When variables change, cross out the old value and put the new value underneath.
4. When a function is called:
  - a. Create a box for all parameters and local variables of the function.
  - b. If a parameter is a scalar, move the value of each actual value into the box for the corresponding formal parameter.
  - c. If a parameter is a list, draw an arrow from the formal parameter box to the box for the corresponding actual parameter.
  - d. Begin execution of the function statements. If a list parameter is modified, follow the arrow to actually change the values.
  - e. When reaching a return statement in the function, move the return value into the variable in the caller that catches the return value.

### Example (right box shows final state of execution)

```
def totalList (list):
    total= 0
    i=0
    while i<len(list):
        total=total+list[i]
        i=i+1
    return total
```

```
# main
scores=[2,4]
total=totalList(scores)
print total
```



**Worksheet**

1. Circle and label the following elements in the code below: formal parameters, actual parameters, object-oriented function calls, function signatures.
2. Trace the code below.

```
def initList1(n):  
  
    i=0  
    list=[]  
    while i<n:  
  
        list.append(i)  
        i=i+1  
  
    return list  
  
def initList2(list,n):  
  
    i=0  
    while i<n:  
  
        list.append(i)  
        i=i+1  
  
# main  
  
list = initList1(2)  
print list  
initList2(list,3)  
print list  
anotherList=[]  
initList2(anotherList,2)  
print anotherList  
list2 = initList1(2)
```