Python Programming

Hellenic Mediterranean University

Lecture 6

Dr. Alina Eqtami

Lecture Roadmap

Σήμερα:

- Advanced use of functions
- Functions calling functions
- Numerical examples
- print and formatted output
- Processing patterns with functions

Motivation

Στόχος:

σπάμε ένα πρόβλημα σε μικρές functions

Example: Modular Design

```
def square(x):
    return x*x

def cube(x):
    return x*x*x

print(square(4))
print(cube(3))
```

Function Composition

```
def square(x):
    return x*x

def f(x):
    return square(x)+2

print(f(5))
```

Nested Function Calls

```
def add(a,b):
    return a+b

def double(x):
    return add(x,x)

print(double(7))
```

More Complex Composition

```
def square(x):
    return x*x

def polynomial(x):
    return square(x)+3*x+2

print(polynomial(4))
```

Example: Statistics

```
def stats(numbers):

    total=0

    for n in numbers:
        total+=n

    avg=total/len(numbers)

    return total,avg

s,m = stats([2,5,7])

print(s,m)
```

Maximum as Function

```
def my_max(data):

    m=data[0]

    for x in data:
        if x>m:
            m=x

    return m

print(my_max([5,9,2,11]))
```

Combining Functions

```
def my_sum(data):

    s=0

    for x in data:
        s+=x

    return s


def average(data):
    return my_sum(data)/len(data)

print(average([4,8,10]))
```

Distance Formula

```
def distance(x1,x2):

   d=x2-x1

   if d<0:
      d=-d

   return d

print(distance(12,4))
```

Energy Example

```
def kinetic(m,v):
   return 0.5*m*v*v

print(kinetic(10,3))
```

Loan Interest

```
def final_amount(P,r):

   return P*(1+r)

print(final_amount(1000,0.05))
```

Printing Results Nicely

Η print δεν είναι μόνο debugging.
Μπορεί να παρουσιάζει αποτελέσματα.

Basic Formatting

```
x=5
y=7

print("x =",x)
print("sum =",x+y)
```

Formatted Strings

```
name="Maria"
score=9.4

print(f"Student: {name}")
print(f"Score: {score}")
```

Using Expressions

```
a=5
b=8

print(f"Sum = {a+b}")
print(f"Product = {a*b}")
```

Function + f-string

```
def square(x):
   return x*x

n=6

print(f"Square of {n} is {square(n)}")
```

Formatted Numerical Output

```
pi=3.1415926

print(f"pi={pi:.2f}")
print(f"pi={pi:.4f}")
```

Count Positives

```
def count_positive(data):

    c=0

    for x in data:
        if x>0:
            c+=1

    return c

print(count_positive([3,-2,5,-1]))
```

Sum of Squares

```
def sum_squares(data):

    s=0

    for x in data:
       s+=x*x

    return s

print(sum_squares([1,2,3]))
```

Function Reuse

```
def square(x):
   return x*x

def sum_squares(data):

   s=0

   for x in data:
      s+=square(x)

   return s
```

Trace 1

```
def f(x):
   return x+2

def g(x):
   return f(x)*3

print(g(4))
```

Τι τυπώνει;

Trace 2

```
def add(a,b):
   return a+b

print(add(2,add(3,4)))
```

Τι τυπώνει;

Common Mistake

```
def square(x):
   print(x*x)

y=square(4)

print(y)
```

Τι συμβαίνει;

Key Ideas

Σήμερα:

- Functions as building blocks
- Composition of functions
- Processing with functions
- print and f-strings
- structured output