ECE12: Introduction to Programming
Lecture 13

Rainer Dömer

doemer@uci.edu

The Henry Samueli School of Engineering
Electrical Engineering and Computer Science
University of California, Irvine

Lecture 13: Overview

- Recursion
  - Recursive function call
  - Concept of recursion
  - Recursion vs. iteration
  - Examples
    - Factorial function
    - Fibonacci series
    - Directory tree traversal

Recursion

- Recursive function
  - function that calls itself
    - directly
    - indirectly
- Concept of recursion
  - Trivial base case
    - return value defined for simple case
    - Example: **`if arg == 0: return 1`**
  - Recursion step
    - reduce the problem towards the base case
    - make a recursive function call
    - Example: **`if arg > 0: return ...fct(arg-1)...`**
- Termination of recursion
  - Converging of recursive calls to the base case
  - Recursive call must be simpler than current call

```
def f():
    ...
    f()
    ...
```

```
def a():
    ...
    b()
    ...

def b():
    ...
    a()
    ...
```

Recursion

- Example: Factorial function (*!*)
  - The factorial of a non-negative integer is
    - *n! = n * (n-1) * (n-2) * (n-3) * ... * 1*
  - This can be written as
    - *n! = n * ((n-1) * ((n-2) * ((n-3) * (... * 1))))*
  - Recursive definition:
    - *1! = 1*
    - *n! = n * (n-1)!*
  - Example computation:

```
5! = 5 * 4!
   = 5 * (4 * 3!)
   = 5 * (4 * (3 * 2!))
   = 5 * (4 * (3 * (2 * 1!)))
   = 5 * (4 * (3 * (2 * 1)))
   = 5 * (4 * (3 * 2))
   = 5 * (4 * 6)
   = 5 * 24
   = 120
```

Recursion

- Example: Factorial function (*!*)
  - Recursive implementation:
    - Base case: *n = 1 : n! = 1*
    - Recursion step: *n > 1 : n! = n * (n-1)!*

```
% python
>>> def factorial(n):
...     if n == 1:
...         return 1
...     else:
...         return n * factorial(n-1)
...
>>> factorial(5)
120
>>> for i in range(1, 10):
...     print factorial(i),
...
1 2 6 24 120 720 5040 40320 362880
```

Recursion vs. Iteration

- Example: Factorial function (*!*)
  - Iterative implementation:
    - Compute *n* products in a loop
    - *n! = n * (n-1) * (n-2) * ... * 1*

```
% python
>>> def factorial_iter(n):
...     product = n
...     for factor in range(n-1, 1, -1):
...         product *= factor
...     return product
...
>>> factorial_iter(5)
120
>>> for i in range(1, 10):
...     print factorial_iter(i),
...
1 2 6 24 120 720 5040 40320 362880
```

Recursion

- Example 2: Fibonacci series
  - Sequence of integers
    - 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233, 377, 610, 987, ...
  - Properties
    - begins with 0 and 1
    - every subsequent Fibonacci number is the sum of the previous two Fibonacci numbers
  - Ratio of successive Fibonacci numbers
    - converging to constant value 1.618...
    - called *Golden Ratio* or *Golden Mean*
  - Recursive definition:
    - Base case: *fibonacci(0) = 0*
fibonacci(1) = 1
    - Recursion step: *fibonacci(n) = fibonacci(n-1) + fibonacci(n-2)*

Recursion

- Example 2: Fibonacci series
  - Recursive implementation:

```
# fibonacci.py: compute the Fibonacci series
# author: Rainer Doemer
#
# modifications:
# 02/12/04 RD   initial version

# function definition
def fibonacci(n):
    if n <= 1:
        return n
    else:
        return fibonacci(n-1) + fibonacci(n-2)

# input, compute, output
while 1:
    i = int(raw_input("Enter an integer (negative to quit): "))
    if i < 0:
        break;
    f = fibonacci(i)
    print "Fibonacci(%d) = %d" % (i,f)
```

Recursion

- Example 3: Directory tree traversal
  - Tree traversal is one of the most important examples for use of recursion
    - Graph theory
    - Depth-First Search (DFS) algorithm
  - In Unix, files are organized in a hierarchy called a directory tree
    - files: single file (base case)
    - directories: list of files (recursion step)
  - Required tools
    - Python module **`os`**: provides operating system functions
    - **`os.path.isdir(name):`** check if **`name`** is a directory (or file)
    - **`os.listdir(name):`** get list of files in directory **`name`**
    - **`os.chdir(name):`** change current directory to **`name`**

Recursion

- Example 3: Directory tree traversal
  - Recursive implementation:

```
# filetree.py: recursively list all files in a directory tree
# author: Rainer Doemer
# 02/13/04 RD   initial version

import os       # use operating system module

# function definition
def list_file(name):
    if os.path.isdir(name):     # is name a directory?
        print "Dir: ", name     # print a directory name
        files = os.listdir(name)# obtain directory entries
        os.chdir(name)          # enter the directory
        for file in files:      # handle each file in directory
            list_file(file)     # recursion!
        os.chdir("..")          # leave the directory
    else:
        print "File:", name     # base case: print a file name

# function call
list_file(".")  # start listing from the current directory
```