RECURSION
Unit 3

CpE 1202L: Data Structures and Algorithm

COMPUTER ENGINEERING
UNIVERSITY of SAN CARLOS

Repetitive Algorithm

Iteration

loop based repetitions of a process

Recursion

repetitive process in which an algorithm calls itself

Recursion

- particularly powerful technique in mathematical definitions
- subprogram may also reference itself
- recursive, it partially consists or is define in term of itself

Recursion

- recursive program P can be expressed as composition P of set of statements S (containing P) and P itself
  - $P \equiv P[S,P]$
  - procedure P contains an explicit reference to itself , then it is said to be directly recursive
  - procedure P contains a reference to another procedure Q, which contains (directly or indirectly) reference to P, P is said to be indirectly recursive

Function is defined recursively

- **anchor or base case**
  - which value of the function is specified for one or more values of the parameter(s)

- **inductive or recursive step**
  - which the functions value for the current value of parameter(s) is defined in terms of previously defined function values and/or parameter values

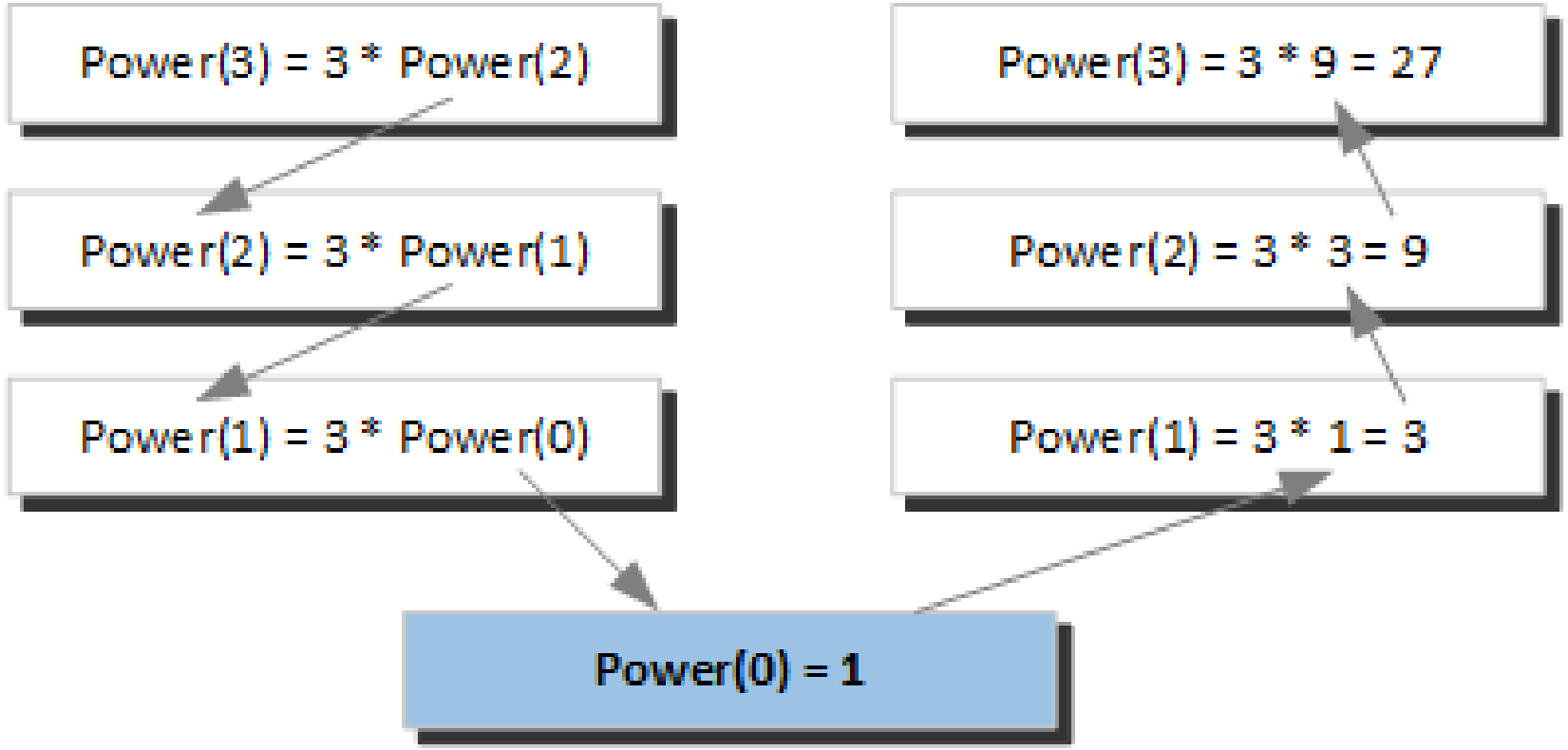
Power Function – A Case Study

- iterative (non-recursive)
  $x^n = x*x*x*$

- sequence of consecutive powers of some #s:
  $3^0 = 1$
  $3^1 = 3$
  $3^2 = 3*3 = 9$
  $3^3 = 3*3*3 = 27$
  $3^4 = 3*3*3*3 = 81$

- $3^4 = 3*3^3 = 3*27 = 81$
  $3^5 = 3*3^4 = 3*81 = 243$

Power Function – A Case Study



Power Function – A Case Study

- need to know
  $3^0 = 1$

- fundamental relation between one power of 3 & the next
  $3^n = 3 * 3^{n-1}$

- recursive definition of power function
  $x^0 = 1$ (anchor or base case)
  for $n>0$, $x^n = x * x^{n-1}$ (inductive or recursive step)

Power Function – A Case Study

Algorithm iterativePower (b, n)

Calculates the power of a number using a loop.

Pre b is the base, n is the number to be raised in power function

Post b^n is returned

1 set i to 0

2 set powerN to 1

3 loop (i < n)

    1 set powerN to powerN * b

    2 increment i

4 end loop

5 return powerN

end iterativePower

Power Function – A Case Study

Algorithm recursivePower (b, n)

Calculates the power of a number using recursion.

Pre b is the base, n is the number to be raised in power function

Post b^n is returned

```
1 if (n equals 0)
        1 return 1
2 else
        1 return (b * recursivePower (b, n - 1))
3 end if
```

end recursivePower

Recursion	Iteration
Recursive function – is a function that is partially defined by itself	Iterative Instructions –are loop based repetitions of a process
Recursion Uses selection structure	Iteration uses repetition structure
Infinite recursion occurs if the recursion step does not reduce the problem in a manner that converges on some condition.(base case)	An infinite loop occurs with iteration if the loop-condition test never becomes false
Recursion terminates when a base case is recognized	Iteration terminates when the loop-condition fails
Recursion is usually slower then iteration due to overhead of maintaining stack	Iteration does not use stack so it's faster than recursion
Recursion uses more memory than iteration	Iteration consume less memory
Infinite recursion can crash the system	infinite looping uses CPU cycles repeatedly
Recursion makes code smaller	Iteration makes code longer

COMPUTER ENGINEERING

UNIVERSITY *of* SAN CARLOS