CMSC21

FUNDAMENTALS OF PROGRAMMING

Kristine Bernadette P. Pelaez

Institute of Computer Science
University of the Philippines Los Baños

RECURSION

RECURSION

A programming technique

RECURSION

express a problem in
smaller instances of itself

RECURSION

In programming.
it is done by
a function calling itself

WHY RECURSION?

reduces unnecessary
function calls

WHY RECURSION?

simplicity

WHY RECURSION?

simplicity

Tree Algorithms
Searching Algorithms

EXAMPLE: EXPONENTIATION

$$b^n$$

EXAMPLE: EXPONENTIATION

$$b^n$$

$$= b * b^{n-1}$$

EXAMPLE: EXPONENTIATION

$$b^n$$

$$= b * b^{n-1}$$

$$= b * b * b^{n-2}$$

EXAMPLE: EXPONENTIATION

$$b^n$$

$$= b * b^{n-1}$$

$$= b * b * b^{n-2}$$

$$= b * b * b * b^{n-3}$$

EXAMPLE: EXPONENTIATION

$$b^n$$

$$= b * b^{n-1}$$

$$= b * b * b^{n-2}$$

$$= b * b * b * b^{n-3}$$

$$= b * b * b * \ldots * b^1$$

TWO CASES IN A RECURSIVE FUNCTION

Base Case

Recursive Case

TWO CASES IN A RECURSIVE FUNCTION

the **relationship** of the smaller problem to the bigger problem

Recursive Case

TWO CASES IN A RECURSIVE FUNCTION

the step that keeps
repeating in the process
until it reach the smallest problem

Recursive Case

EXAMPLE: EXPONENTIATION

$$b^n$$

$$= b * b^{n-1}$$

$$= b * b * b^{n-2}$$

$$= b * b * b * b^{n-3}$$

$$= b * b * b * \ldots * b^1$$

TWO CASES IN A RECURSIVE FUNCTION

Base Case

aka the terminating condition.
this is the **smallest subproblem** that can be reached by the recursive case

EXAMPLE: EXPONENTIATION

$$b^n$$

$$= b * b^{n-1}$$

$$= b * b * b^{n-2}$$

$$= b * b * b * b^{n-3}$$

$$= b * b * b * \ldots * b^1$$

RECURRENCE RELATION

an **equation** that is defined by itself

RECURRENCE RELATION

used to represent
recursive processes

RECURRENCE RELATION

$$b^n = b * b * b * \ldots * b^1$$

RECURRENCE RELATION

$$b^n = f(n, b)$$

$$f(n, b) = \begin{cases} \text{recursive case(s)} \\ \text{base case(s)} \end{cases}$$

RECURRENCE RELATION

$$b^n = f(n, b)$$

$$f(n, b) = \begin{cases} b & , \quad n=1 \\ \end{cases}$$

RECURRENCE RELATION

$$b^n = f(n, b)$$

$$f(n,b) = \begin{cases} b & , \quad n=1 \\ b * b^{n-1} & , \quad n>1 \end{cases}$$

RECURRENCE RELATION

$$b^n = f(n, b)$$

$$f(n,b) = \begin{cases} b & , \quad n=1 \\ b * b^{n-1} & , \quad n>1 \end{cases}$$

RECURRENCE RELATION

$$b^n = f(n, b)$$

$$f(n,b) = \begin{cases} b & , \ n=1 \\ b*f(n-1,b) & , \ n>1 \end{cases}$$

RECURSIVE FUNCTION

```
int f(){

}
```

$$f(n,b) = \begin{cases} b & , \ n=1 \\ b*f(n-1,b), & n>1 \end{cases}$$

RECURSIVE FUNCTION

```
int f(int n, int b){

}
```

$$f(n,b) = \begin{cases} b & , \ n=1 \\ b*f(n-1,b), & n>1 \end{cases}$$

RECURSIVE FUNCTION

```
int f(int n, int b){
   if(n==1){
      return b;
   }
}
```

$$f(n,b) = \begin{cases} b & , \ n=1 \\ b*f(n-1,b), & n>1 \end{cases}$$

RECURSIVE FUNCTION

```
int f(int n, int b){
  if(n==1){
    return b;
  }
  else if(n>1){

  }
}
```

$$f(n,b) = \begin{cases} b & , \ n=1 \\ b*f(n-1,b), & n>1 \end{cases}$$

RECURSIVE FUNCTION

```
int f(int n, int b){
  if(n==1){
    return b;
  }
  else if(n>1){
    return b*f(n-1,b);
  }
}
```

$$f(n,b) = \begin{cases} b & , \ n=1 \\ b*f(n-1,b), & n>1 \end{cases}$$

EXAMPLE

DEMO

NEXT MEETING

SECRET :)

CMSC21

FUNDAMENTALS OF PROGRAMMING

Kristine Bernadette P. Pelaez

Institute of Computer Science
University of the Philippines Los Baños