NC STATE UNIVERSITY

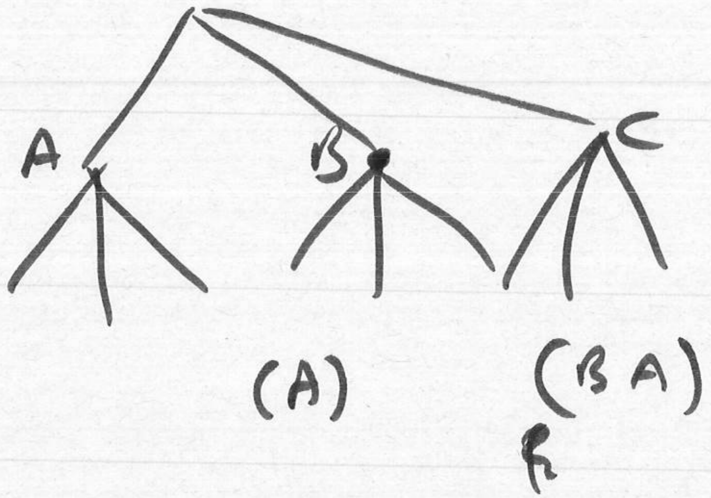
ENGINEERING ONLINE

# Lecture Notes

| | |
|---|---|
| **Course Number:** | CSC 513 |
| **Instructor:** | Dr. Singh |
| **Lecture Number:** | 24 |

nextsib

prevsib



1.. last()

... prevsib ($x) {

node()

$x/ preceding-sibling :: element() [last()]

[1]

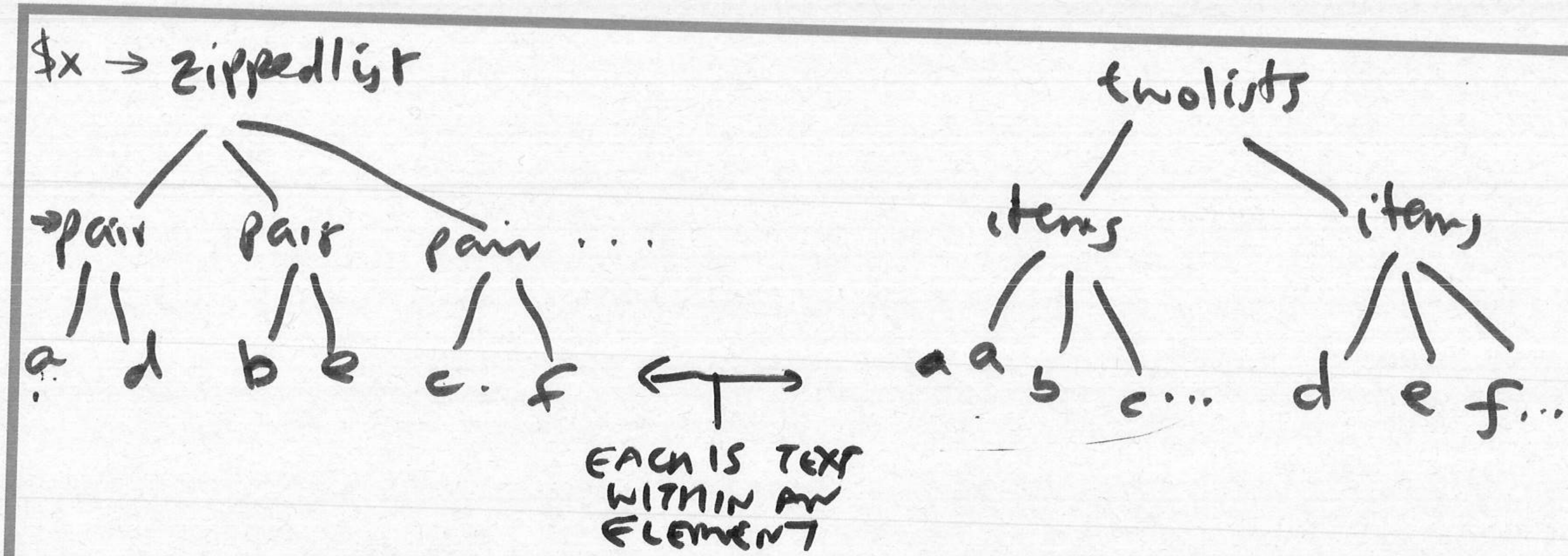
FROM ~~ZU~~ TWOLISTS TO ZIPPEDLIST

```
zip ($t){
<zippedlist>
         $i...1 to (sizeof firsts)
    for
      let $first :=  $firsts [$i]
      let $second :=  $seconds [$i]
    return  <pair>
                $first,
                $second
            </pair>

  </zippedlist>
       }
```

```
[let $firsts := $t/items[1]/~~*~~
[let          :=          [2]
```

```
zip($t){
 <zippedlist>
  let $firsts := $t/items(1)
  let   seconds := $t/items[2]
  for $first in $firsts/item at $i
          let         :=
          ~~for~~ $second in $seconds/item[$i]
  return  <pair>
             $first,
             $second
          </pair>
 </zippedlist>
         }
```



EACH IS TEXT WITHIN AN ELEMENT

unzip ($x) {

~~$x pair~~ ~~*~~

~~$x *~~

for $y in ... ?!

~~$x/*[1]~~

$x/pair/*[1]

return $y

<twolists>

<items>

</items>

<items>

</items>

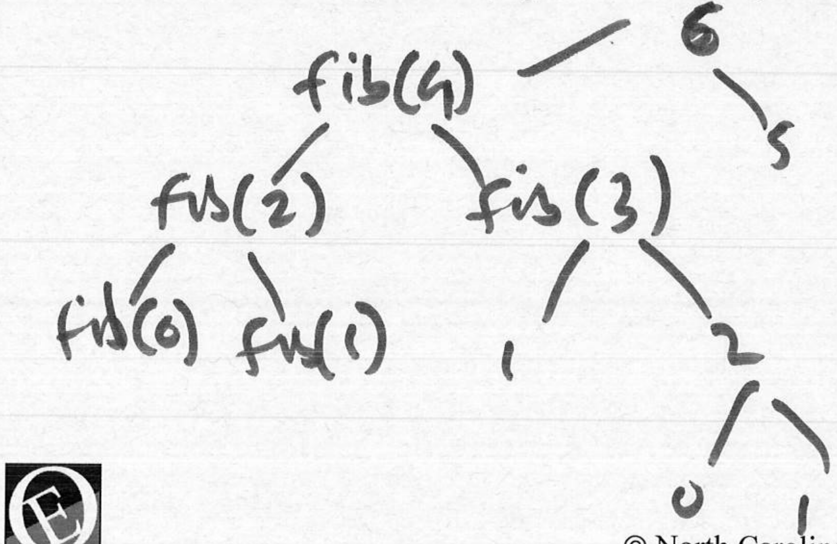
</twolists>

# RECURSIVE

```
fact ($n) {
  if ($n = 0)
  then  1
  else  $n *
        fact ($n-1)
}
```

Complexity of above rec.. ~~above~~ fact : O(n )

iterative : O(n)



# FIBONACCI

fib(n) 1, 1, 2, 3, 5, 8, 13,

n 0 1 2 3 4

```
fib ($n) {
  if (($n = 0) or ($n = 1))
  then 1
  else fib ($n-1)
     + fib ($n-2)
}
```

Complexity of typical iterative fib : O(n )

Complexity of above recursive fib

$(1.6)^n$

© North Carolina State University, All Rights Reserved

CAN WE WRITE A RECURSIVE FIB ALGO OF O(n) COMPLEXITY YET.

O(1) space
O(n) time

```
~~FIB(~~ current =
previous = 1
prevprev = 1
for $i:=0 . .
 { ~~previous~~ =
   prevprev = previous
   previous = current
      :
      :
 }
 return current
```

( Usea an array to store fib(n)

Space O(n)

Want O(1) space

```
fib(
{
  if n=0 or n=1
then return 1
else
```