**Logistics:** Masks are recommended.
- HWD due tonight 9pm
- Gradescope self-enrollment code: WVVEEG

"Describe an algorithm" includes justification
but NOT CODE

"Analyze an algorithm" includes justification

---

Given a string, can we split it into a sequence of "words"

\[
\text{IsWord}(str) \rightarrow \text{bool}
\]

---

PRIMUS
DIGNITAS
INTAMTEN
VIVISCIENTIA
NONPOTEST
ESSERERES
ENIMSVN
PARVAE
PROPEINSINGVLIS
LITTERIS
ATQVE
INTERPVNCTIONIBUSVERBORVMOC
CCVPATAE

sequence of letters $\rightarrow$ sequence of words

- $\text{nothing}$
- $\text{word}$, seq of words

My job

**What is the first word?**

try all prefixes:

Is Word = True?
Is rest splittable into words?

---

**Splittable(A[1..n]):**

if $n = 0$
    return True
for $i \leftarrow 1$ to $n$
    if IsWord(A[1..i])
        if Splittable(A[i+1..n])
            return True
return False
Treat input string $A$ as a global variable

$$\text{Splittable}(i) = \text{True iff } A[i..n] \text{ is splittable into words}$$

$$\text{Splittable}(i) = \begin{cases} 
\text{True} & \text{if } i > n \\
\bigvee_{j=i}^{n} (\text{IsWord}(i, j) \land \text{Splittable}(j + 1)) & \text{otherwise}
\end{cases}$$

\[\langle \text{Is the suffix } A[i..n] \text{ Splittable?} \rangle\]

**SPLITTABLE(i):**

if $i > n$

return True

for $j \leftarrow i$ to $n$

if $\text{IsWord}(i, j)$

if SPLITTABLE$(j + 1)$

return True

return False

\[T(n) = \text{time for SPLITTABLE}(n) = O(2^n)\]

\[T(n) = \sum_{i=1}^{n} (O(1) + T(n-i))\]

Nowhere QFVY...-

A lot of redundant computation

**MEMORIZATION**

Store answers in an array $\text{SplitTable}[1..n]$

Time $\rightarrow O(n^2)$ calls to $\text{IsWord}$

The array fills from right to left
Do that on purpose!

**DPSplitTable(A(1..n))**:

1. SplitTable[n+1] ← TRUE
2. **FOR** i ← 1 **down to** 1
   - SplitTable[i, j] ← FALSE
3. **FOR** j ← i to n
   - if a word(A[i, j])
     - if SplitTable[j+1]
       - SplitTable[i, j] ← TRUE
4. return SplitTable[1]

---

**Longest Increasing Subsequence**

What is the first element of LIS?  
Is A[1] in the LIS?

Increasing sequence bigger than x
　is either ∅  
　or (int y > x) followed by (incseq > y)

Global variable input A[1..n]

\[ \text{LIS}(i, j) = \begin{cases} 0 & \text{if } j > n \\ \text{LIS}(i, j+1) & \text{if } A[i] \geq A[j] \\ \max \{ \text{LIS}(i, j+1), 1 + \text{LIS}(j, j+1) \} & \end{cases} \]
FASTLIS(A[1..n]):
A[0] ← −∞  \hspace{1em}  ⟨\text{Add a sentinel}\rangle
for i ← 0 to n  \hspace{1em}  ⟨\text{Base cases}\rangle
   \hspace{1em}  LISbigger[i, n + 1] ← 0
for j ← n down to 1
   \hspace{1em}  for i ← 0 to j − 1  \hspace{1em}  ⟨\text{...or whatever}\rangle
      \hspace{1em}  keep ← 1 + LISbigger[j, j + 1]
      \hspace{1em}  skip ← LISbigger[i, j + 1]
      \hspace{1em}  if \ A[i] ≥ A[j]
      \hspace{1em}     LISbigger[i, j] ← skip
      \hspace{1em}  else
      \hspace{1em}     LISbigger[i, j] ← \max\{keep, skip\}
return LISbigger[0, 1]