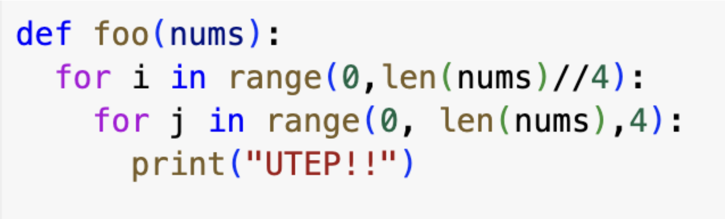
CS 2302 – Data Structures
Exam One Study Guide

Dr. Daniel Mejia
Fall 2024

Note: This is intended to be a study guide. In no way is every potential question on the exam listed in this guide. Please continue to study beyond this study guide. All topics discussed in the course may be written in the exam.

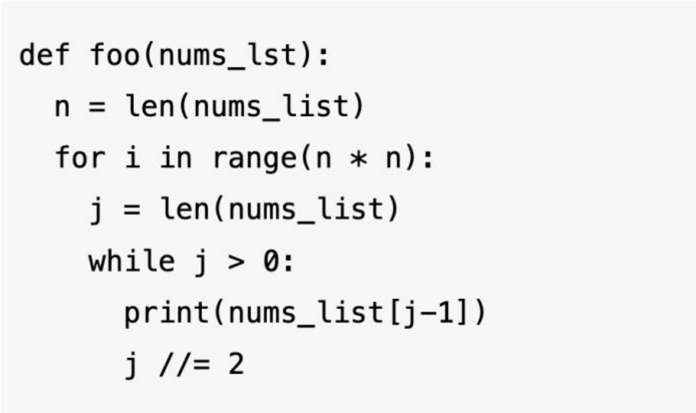
Time Complexity

What is the running time of the following function?



```python
def foo(nums):
  for i in range(0,len(nums)//4):
    for j in range(0, len(nums),4):
      print("UTEP!!")
```

What is the running time of the following function?



```python
def foo(nums_lst):
  n = len(nums_list)
  for i in range(n * n):
    j = len(nums_list)
    while j > 0:
      print(nums_list[j-1])
      j //= 2
```

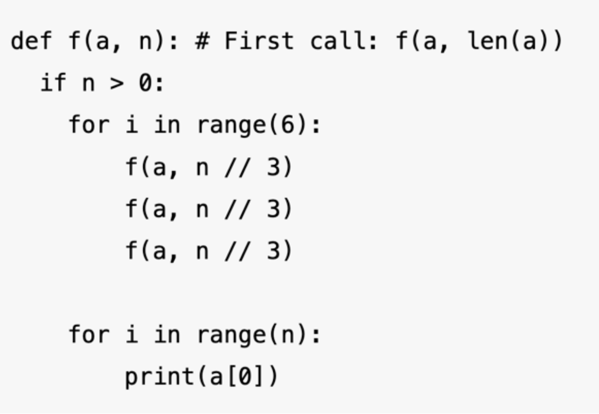
Iterative Method

Consider the following recurrence equation: T(n) = T(n/2) + n. What is T(16)? Solve the equation by iteration.

Consider the following recurrence equation: T(n) = T(n-1) +1. What is the Big-O running time complexity? Solve the equation using the iterative method.

Master Theorem

What is the recurrence equation that describes the running time of the following recursive function? T(n) = a T(n / b) + n ^ k.

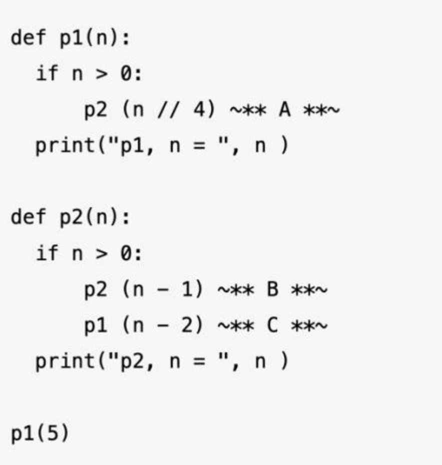


```
def f(a, n): # First call: f(a, len(a))
  if n > 0:
    for i in range(6):
        f(a, n // 3)
        f(a, n // 3)
        f(a, n // 3)

    for i in range(n):
        print(a[0])
```

Recursion Tracing

Trace the execution of the following code using a recursion tree. Every time you create a new node, assign it an ID starting from 0.



```
def p1(n):
  if n > 0:
      p2 (n // 4) ~** A **~
  print("p1, n = ", n )

def p2(n):
  if n > 0:
      p2 (n - 1) ~** B **~
      p1 (n - 2) ~** C **~
  print("p2, n = ", n )

p1(5)
```

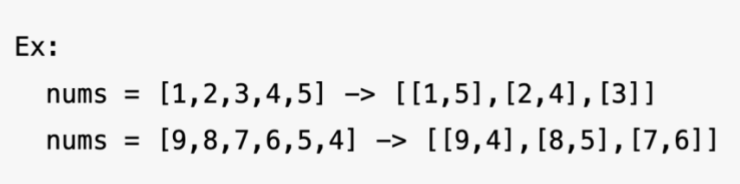
How many nodes were created? (Initial call to p1(5)) is node 0. If you drew 3 nodes, the answer would be 3. If you drew 5 nodes, the answer would be 5, etc.

What is the value of n in node #3?

What is the first line printed on the console?

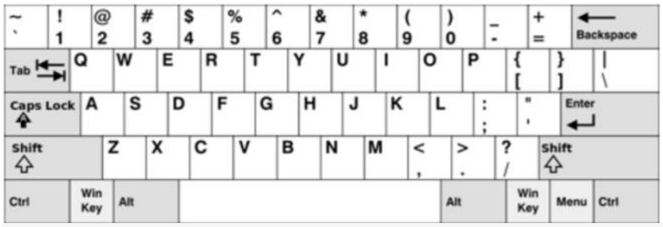
Coding Problems

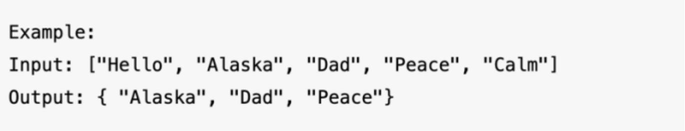
Write a function that takes in a list and creates a list of lists that follows this pattern:



```
Ex:
  nums = [1,2,3,4,5] -> [[1,5],[2,4],[3]]
  nums = [9,8,7,6,5,4] -> [[9,4],[8,5],[7,6]]
```

Given a list of words, return the words that can be typed using letters of the alphabet that appear on ****exactly one**** or ****exactly three**** different rows of the American keyboard (see image below).


~ ` ! 1 @ 2 # 3 $ 4 % 5 ^ 6 & 7 * 8 (9) 0 _ - + = Backspace
Tab Q W E R T Y U I O P { [}] | \
Caps Lock A S D F G H J K L : ; " ' Enter
Shift Z X C V B N M < , > . ? / Shift
Ctrl Win Key Alt Alt Win Key Menu Ctrl

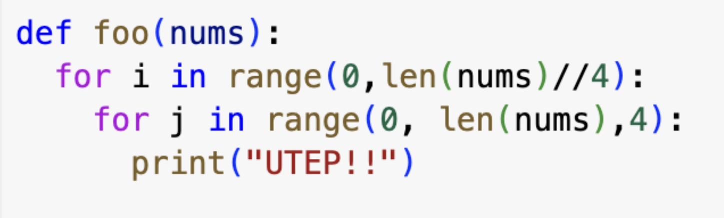


```
Example:
Input: ["Hello", "Alaska", "Dad", "Peace", "Calm"]
Output: { "Alaska", "Dad", "Peace"}
```

SOLUTIONS

Time Complexity

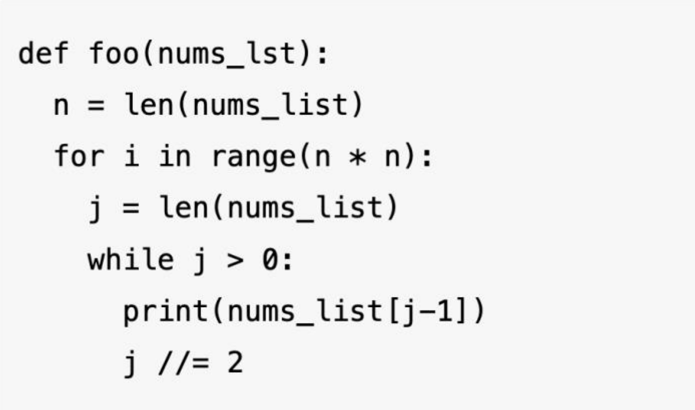
What is the running time of the following function?



```
def foo(nums):
  for i in range(0,len(nums)//4):
    for j in range(0, len(nums),4):
      print("UTEP!!")
```

ANSWER: O(n^2)

What is the running time of the following function?



```
def foo(nums_lst):
  n = len(nums_list)
  for i in range(n * n):
    j = len(nums_list)
    while j > 0:
      print(nums_list[j-1])
      j //= 2
```

ANSWER: O(n^2 log n)

Iterative Method

Consider the following recurrence equation: T(n) = T(n/2) + n. What is T(16)? Solve the equation by iteration.

ANSWER:

T(1) = 1
T(2) = 3
T(4) = 7
T(8) = 15
T(16) = 31

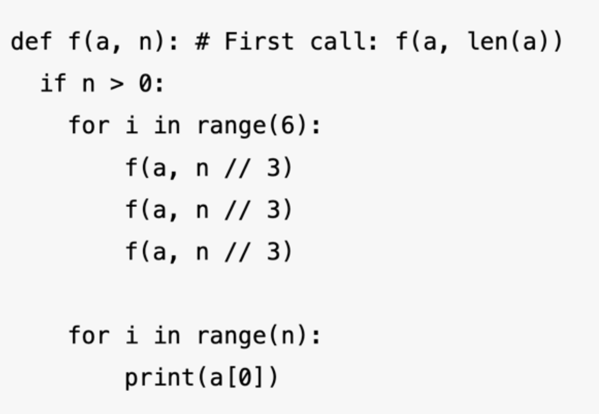
Consider the following recurrence equation: T(n) = T(n-1) +1. What is the Big-O running time complexity? Solve the equation using the iterative method.

ANSWER:

T(1) = 1
T(2) = 2
T(3) = 3
T(4) = 4
T(5) = 5
O(n)

Master Theorem

What is the recurrence equation that describes the running time of the following recursive function? T(n) = a T(n / b) + n ^ k.



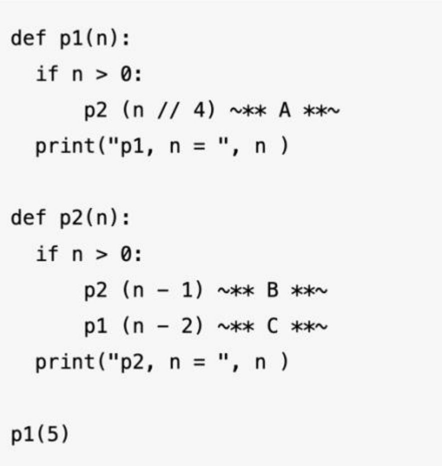
```
def f(a, n): # First call: f(a, len(a))
  if n > 0:
    for i in range(6):
        f(a, n // 3)
        f(a, n // 3)
        f(a, n // 3)

    for i in range(n):
        print(a[0])
```

ANSWER:
a = 18
b = 3
k = 1

Recursion Tracing

Trace the execution of the following code using a recursion tree. Every time you create a new node, assign it an ID starting from 0.

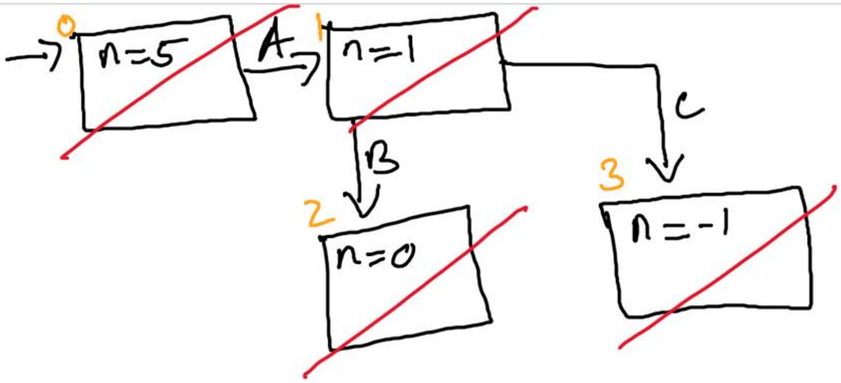


```
def p1(n):
  if n > 0:
      p2 (n // 4) ~** A **~
  print("p1, n = ", n )

def p2(n):
  if n > 0:
      p2 (n - 1) ~** B **~
      p1 (n - 2) ~** C **~
  print("p2, n = ", n )

p1(5)
```

TRACE:



```
p2, n= 0
P1, n=-1
P2, n=1
P1, n=5
```

How many nodes were created? (Initial call to p1(5)) is node 0. If you drew 3 nodes, the answer would be 3. If you drew 5 nodes, the answer would be 5, etc.

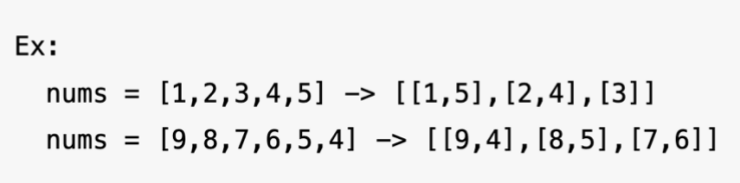
ANSWER: 4

What is the value of n in node #3?
ANSWER: n=-1

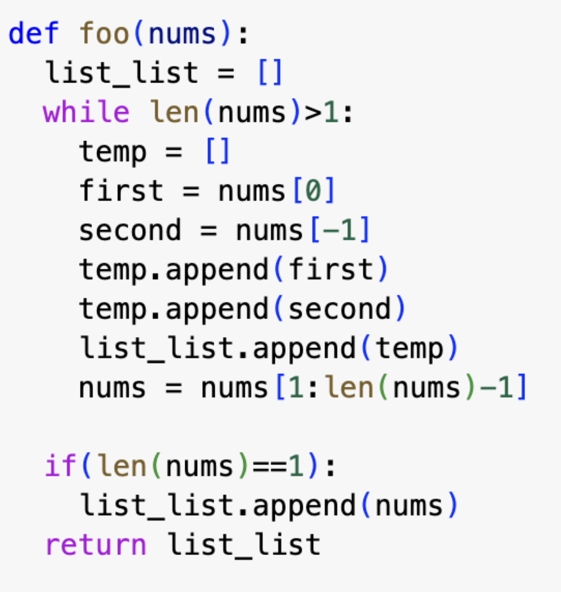
What is the first line printed on the console?
ANSWER: p2, n=0

Coding Problems

Write a function that takes in a list and creates a list of lists that follows this pattern:



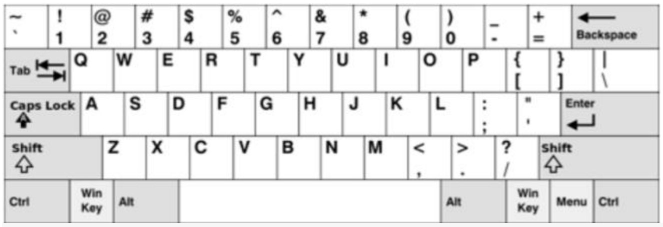
```
Ex:
  nums = [1,2,3,4,5] -> [[1,5],[2,4],[3]]
  nums = [9,8,7,6,5,4] -> [[9,4],[8,5],[7,6]]
```

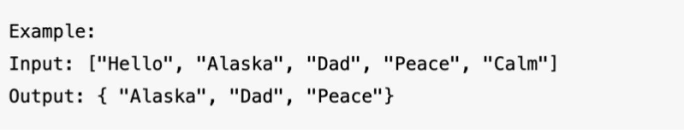


```
def foo(nums):
  list_list = []
  while len(nums)>1:
    temp = []
    first = nums[0]
    second = nums[-1]
    temp.append(first)
    temp.append(second)
    list_list.append(temp)
    nums = nums[1:len(nums)-1]

  if(len(nums)==1):
    list_list.append(nums)
  return list_list
```

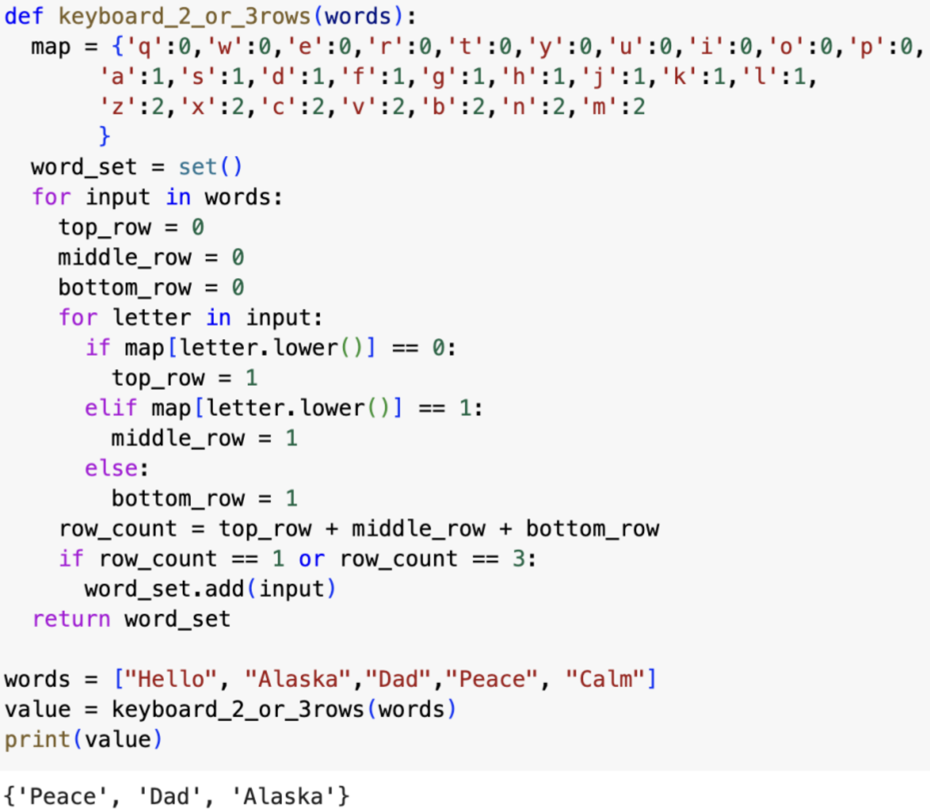
Given a list of words, return the words that can be typed using letters of the alphabet that appear on **exactly one** or **exactly three** different rows of the American keyboard (see image below).


~ ` ! 1 @ 2 # 3 $ 4 % 5 ^ 6 & 7 * 8 (9) 0 _ - + = Backspace
Tab Q W E R T Y U I O P { [}] | \
Caps Lock A S D F G H J K L : ; " ' Enter
Shift Z X C V B N M < , > . ? / Shift
Ctrl Win Key Alt Alt Win Key Menu Ctrl



```
Example:
Input: ["Hello", "Alaska", "Dad", "Peace", "Calm"]
Output: { "Alaska", "Dad", "Peace"}
```

SAMPLE ANSWER:



```
def keyboard_2_or_3rows(words):
  map = {'q':0,'w':0,'e':0,'r':0,'t':0,'y':0,'u':0,'i':0,'o':0,'p':0,
       'a':1,'s':1,'d':1,'f':1,'g':1,'h':1,'j':1,'k':1,'l':1,
       'z':2,'x':2,'c':2,'v':2,'b':2,'n':2,'m':2
       }
  word_set = set()
  for input in words:
    top_row = 0
    middle_row = 0
    bottom_row = 0
    for letter in input:
      if map[letter.lower()] == 0:
        top_row = 1
      elif map[letter.lower()] == 1:
        middle_row = 1
      else:
        bottom_row = 1
    row_count = top_row + middle_row + bottom_row
    if row_count == 1 or row_count == 3:
      word_set.add(input)
  return word_set

words = ["Hello", "Alaska","Dad","Peace", "Calm"]
value = keyboard_2_or_3rows(words)
print(value)
```

```
{'Peace', 'Dad', 'Alaska'}
```