UNIVERSITY OF CALIFORNIA, BERKELEY
Department of Electrical Engineering and Computer Sciences
Computer Science Division

CS10 Summer 2023 **Instructor: Vedansh Malhotra**

PRACTICE EXAM

SOLUTIONS

In-Lab Final Exam [XXXX-XXXX hrs]

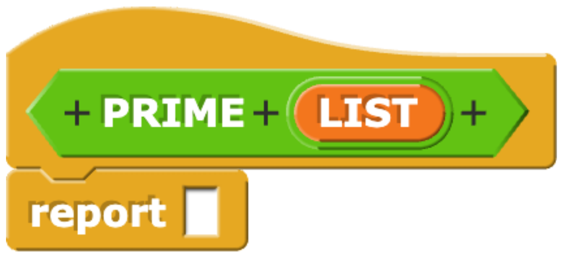
You will sign the following on your actual exam:

As a member of the UC Berkeley community, I act with honesty, integrity, and respect for others.
I have neither received nor given any assistance in taking this exam.
I will not discuss the contents of this exam with anybody before ***XXXX on XXXX****.*

Problem-1: Primality Testing

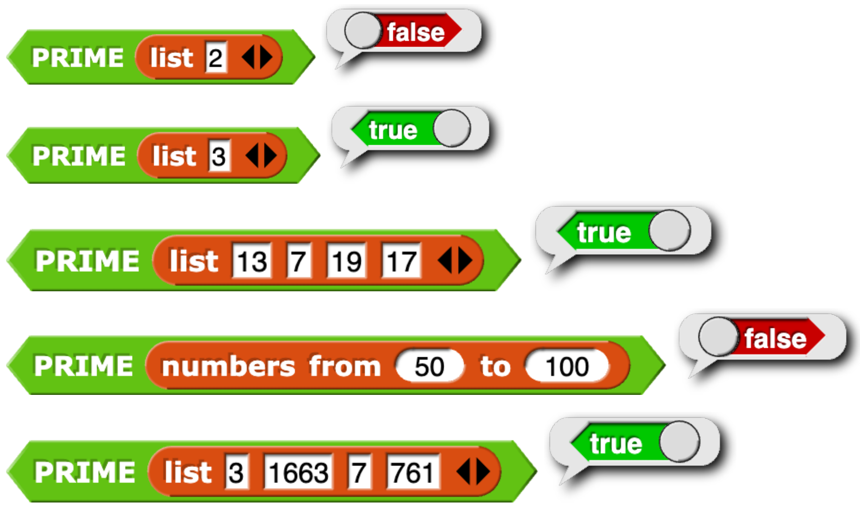
The block **`PRIME`** takes in a list of numbers. The length of the list is at least 1, and the numbers it contains are all >= 2. The block reports `True` if all the numbers in the list are prime (are only divisible by 1 and themselves) and `False` otherwise.

Below, you'll find the provided skeleton code. You may **only** fill out the report block — no other code should be added anywhere. Don't import any libraries or create any other blocks.



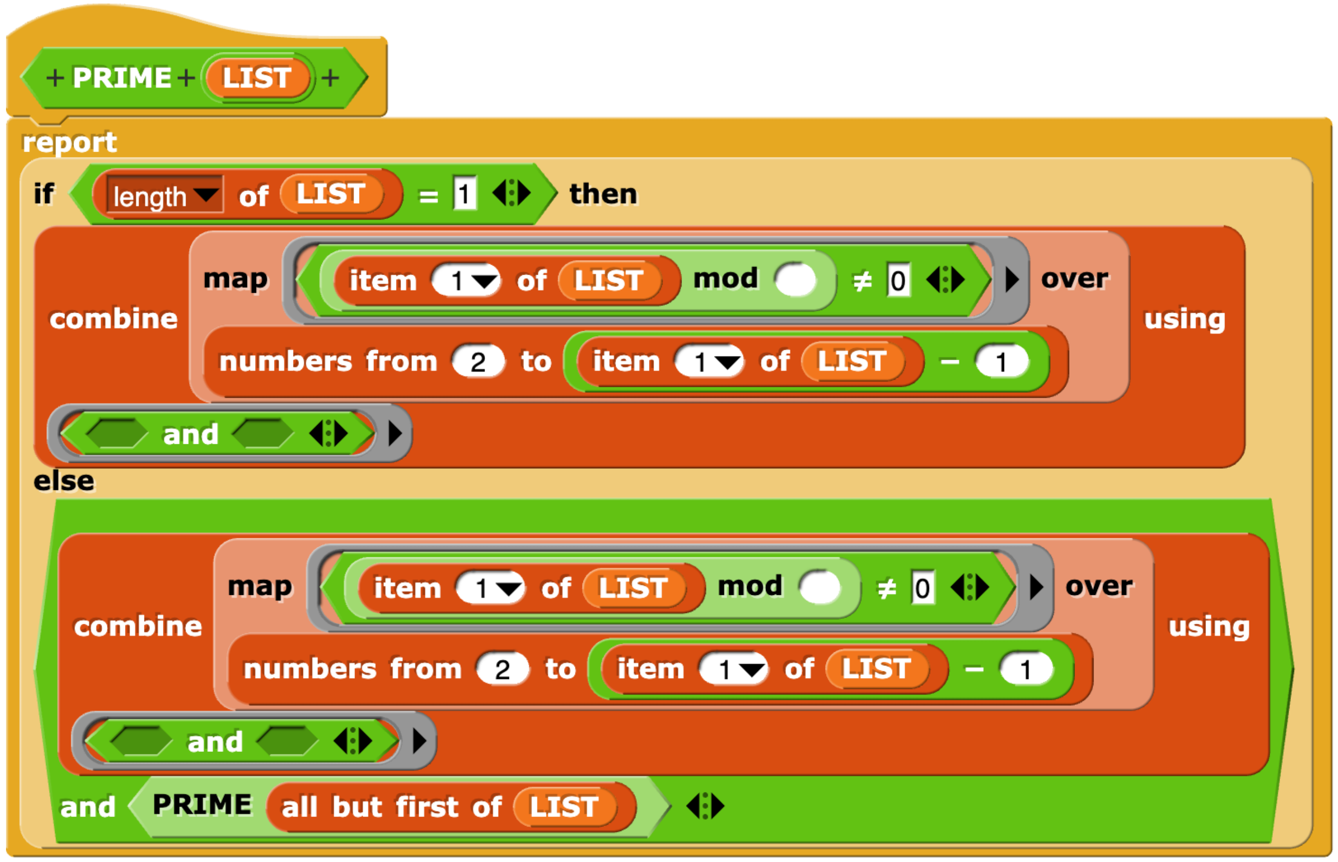
```
+ PRIME + (LIST) +
report [ ]
```

Here are some examples:



```
PRIME (list 2)                      -> false
PRIME (list 3)                      -> true
PRIME (list 13 7 19 17)             -> true
PRIME (numbers from 50 to 100)      -> false
PRIME (list 3 1663 7 761)           -> true
```

SOLUTION:



```
+ PRIME + (LIST) +
report
if (length of LIST) = 1 then
    combine
        map ((item 1 of LIST) mod ( ) ≠ 0) over
            numbers from 2 to ((item 1 of LIST) − 1)
    using (( ) and ( ))
else
    (combine
        map ((item 1 of LIST) mod ( ) ≠ 0) over
            numbers from 2 to ((item 1 of LIST) − 1)
    using (( ) and ( )))
    and (PRIME (all but first of LIST))
```

Problem-2: Numerals

Write a Python function called `numerals` that takes in `b`, an integer >1, and `k`, an integer >= 1, and returns all possible `k`-digit numerals in base `b`.

The function should return a nested list. Each sublist should represent one k-digit numeral, and each item in the sublist should be an integer representing one digit of the numeral. The sublists should be arranged in ascending order.

Don't write any helper functions, and don't import anything.

Here are some examples:

```
>>> numerals(2, 1)
[[0], [1]]
>>> numerals(2, 2)
[[0, 0], [0, 1], [1, 0], [1, 1]]
>>> numerals(2, 3)
[[0, 0, 0], [0, 0, 1], [0, 1, 0], [0, 1, 1], [1, 0, 0], [1,
0, 1], [1, 1, 0], [1, 1, 1]]
>>> numerals(3, 1)
[[0], [1], [2]]
>>> numerals(3, 2)
[[0, 0], [0, 1], [0, 2], [1, 0], [1, 1], [1, 2], [2, 0], [2,
1], [2, 2]]
```

SOLUTION:

```
def numerals(b, k):
    """
    Returns a list of lists containing all possible numerals
with k > 0 digits in base b > 1.
    """

    digits = [[i] for i in range(b)]

    if k == 1:
        return digits

    else:
        recurse = numerals(b, k-1)
        result = []
        for item in recurse:
            for digit in digits:
                result += [item + digit]

        return result
```

Problem-3: OOP

This problem was adapted from the Lecture-17 Quiz. This version is more reflective of the style of problems we typically use on exams.

Your task is to fill out the starter code linked here (bit.ly/3O7DHGy), such that all the doctests pass. You can download the file, open it in code.cs61a.org, and then run the doctests by clicking the red test-tube shaped button in the top right corner.

SOLUTION: Linked here (bit.ly/3Y1YKim)