EPFL CIVIL-127, Lab 3 solution

3.1

```python
# Sieve of Eratosthenes implementation
# See https://en.wikipedia.org/wiki/Sieve_of_Eratosthenes

N = 272
# False => number is prime
# True => number is composite
a = [False] * N
for i in range(2, N):
    if not a[i]:
        print(i, end=" ")
        for j in range(i+i, N, i):
            a[j] = True
```

3.2

```
# Simulate lots of Snakes & Ladders games and calculate average
# number of moves per game.
# Short implementation, using dict.

import random

snakes_and_ladders = {
    43:  17, 50:  5, 56:  8, 73:  15, 84:  63, 87:  49, 98:  40,
    2:  23, 6:  45, 20:  59, 52:  72, 57:  96, 71:  92, 101: 99,
    102: 40, 103: 97, 104: 96, 105: 95,
}
moves = 0
N = 1_000_000

for i in range(N):
    marker = 1
    while marker != 100:
        marker += random.randint(1, 6)
        moves += 1
        marker = snakes_and_ladders.get(marker, marker)

print("average: ", moves / N)
```

Another solution would be to use a list. As long as the number of `if` statements is reduced and you produce the correct answer in reasonable time, you are good.

3.3

When using `*` to expand a list, the elements are not copied. The list references the elements multiple times. This behavior often haunts new programmers; make sure you properly understand what's going on.

We can create a 5x8 grid by using a `for` loop. Inside each iteration, a new array of 8 elements is created.

```
grid = []
for i in range(5):
    grid.append([0] * 8)
print(grid)

grid[4][2] = 99
print(grid)
```

3.4

```
# Code to solve N people in a circle puzzle (Josephus problem)
# See https://en.wikipedia.org/wiki/Josephus_problem

N = 5
K = 2

active = K  # which person will be removed next
people = list(range(N))

while len(people) > 1:
    print(people)
    print("removing", people[active])
    people = people[:active] + people[active+1:]
    active = (active + K - 1) % len(people)
    print()

print("The answer is:", people[0])
```

3.5

```
# Bruteforce SEND+MORE=MONEY puzzle

for s in range(1, 10):  # we start at 1 for s
    for e in range(0, 10):
        for n in range(0, 10):
            for d in range(0, 10):
                for m in range(1, 10):  # we start at 1 for m
                    for o in range(0, 10):
                        for r in range(0, 10):
                            for y in range(0, 10):
                                # check that we have unique values by
                                # creating a set and checking its len
                                values = {s, e, n, d, m, o, r, y}
                                if len(values) == 8:
                                    send = 1000*s + 100*e + 10*n + d
                                    more = 1000*m + 100*o + 10*r + e
                                    money = 10000*m + 1000*o + 100*n + 10*e + y
                                    if send + more == money:
                                        print(send)
                                        print(more)
                                        print("=====")
                                        print(money)
```

We create a whole bunch of loops and use a `set` to check that we have 8 distinct values.

We could also write a very long `if` statement but that's painful to read and write (28 comparisons!):
`if s!=e and s!=n and ... and s!=y and e!=n and ...`

The answer you should get:
9567
1085
=====
10652