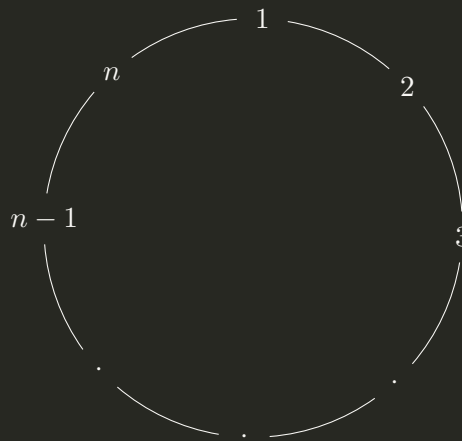




Assignment V: Links

(Due on On Léa)


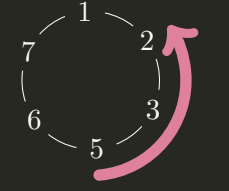

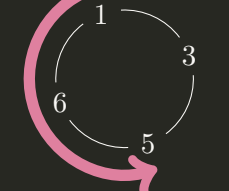

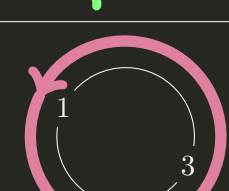
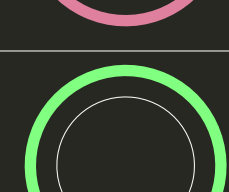
Imagine that there are n people in a circle, numbered $1, 2, \dots, n$:



Starting from the first person (number 1), count clockwise around the circle, to the m -th person and remove them from the circle. The circle closes up when a person is removed. Now, count counter-clockwise around the circle to the o -th person from where you were, and remove that person.

Repeat this process of counting and removing, switching between the clockwise and counter-clockwise directions and counts. Who is the last person in the circle and what is the order in which they are removed?

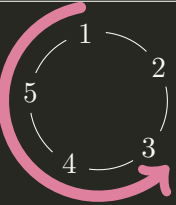
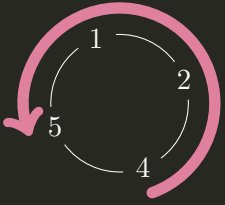
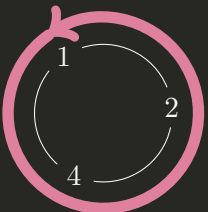
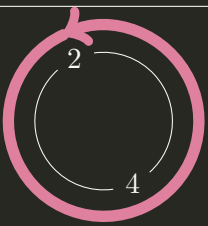
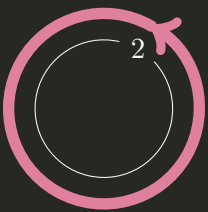
Example: $(n = 7, m = 4, o = 3)$

Circle	Current Position	Count	Remove
	1	1 2 3 4	4
	5	5 3 2	2
	3	3 5 6 7	7
	1	1 6 5	5
	6	6 1 3 6	6
	1	1 3 1	1
	3	3 3 3 3	3

Example ($n = 5, m = 3, o = 0$):

Circle	Current Position	Count	Remove
	1	1 2 3	3
	4	4 5 1	1
	2	2 4 5	5
	2	2 4 2	2
	4	4 4 4	4

Example ($n = 5, m = 0, o = 4$):

Circle	Current Position	Count	Remove
	1	1 5 4 3	3
	4	4 2 1 5	5
	1	1 4 2 1	1
	2	2 4 2 4	4
	2	2 2 2 2	2

Implementation

Write a python program to solve this problem using the circular link chain programming technique. Specifically, you should use a *doubly*-linked chain. Use the following definition of a link:

```
class DoubleLink(Generic[T]):
    def __init__(self, element: Optional[T] = None):
        self.element: Optional[T] = element
        self.next: Optional[DoubleLink[T]] = None
        self.prev: Optional[DoubleLink[T]] = None
```

Class Circle

Start from the class `Circle` in the file `circle.py`. You can add additional methods but must at least implement the following two operations:

Signature	<code>def remove_next(self) → int</code>
Description	Remove the next person (only one!) from the circle according to the rules.
Pre-conditions	Circle is created.
Mutator	Yes
Returns	The number (name) of the person removed.

Signature	<code>def print_circle(self)</code>
Description	Prints the numbers (names) of the people currently in the circle. Starts from the current counting position and proceeds clockwise. Format is one-line, comma separated. Ex: circle is now 4, 5, 1, 2.
Pre-conditions	Circle is created
Mutator	No
Returns	None

Requirements

Your program *must* meet the following requirements:

1. Implement the circle in the file `circle.py`. Implement the two methods outlined in the previous section.
2. Create a `main.py` that will take user input and print the results.

-
3. Read the integers n , m and o from standard input (console). Only accept valid values, that is

- $n > 0$
- $m, o \geq 0$
- $m + o > 0$ (at least remove in one direction).

Re-prompt the user for input of any of the above are false.

4. Print the order or people removed to standard output (console). Your output must look like:

```
n> 7
m> 4
o> 3
circle: 1, 2, 3, 4, 5, 6, 7

4 is removed
circle now is 5, 6, 7, 1, 2, 3

2 is removed
circle now is 3, 5, 6, 7, 1

7 is removed
circle now is 1, 3, 5, 6

5 is removed
circle now is 6, 1, 3

6 is removed
circle now is 1, 3

1 is removed
circle now is 3

3 is removed
```

Submission

Once you've made your final `git push` to GitHub, submit a text file with the commit id to LEA.

Style

Your program should be clear and well commented. It must follow the Coding Standards as given in the class notes