CSC148: Stacks and Queues

Daniel Zingaro
daniel.zingaro@utoronto.ca

Question 1

Consider the following code.

```
def change(s1: Stack) -> None:
  s2 = Stack()
  while not s1.is_empty():
    first = s1.pop()
    if s1.is_empty():
      s2.push(first)
    else:
      second = s1.pop()
      s2.push(second)
      s2.push(first)
  while not s2.is_empty():
    s1.push(s2.pop())
```

1. Given the below stack `s`, what are the contents of `s` after running `change(s)`?

```
s = Stack()
s.push(1)
s.push(2)
s.push(3)
s.push(4)
change(s)
```

2. Complete the following sentence:
If `change` is run on a stack `s`, then `s` changes so that ... (i.e. we want a brief, English description of what the function does, not a line-by-line explanation of the code).

Question 2

Write function `roll` below. `roll` takes the element at position `n` on the stack and makes it the top element. (The top of stack is position 1, the next is position 2, and so on.)

For example, consider stack `s` as A B C D, where the left end is the bottom of the stack and the right end is the top (so D is at the top of the stack). `roll(s, 2)` will yield A B D C and `roll(s, 3)` will yield A C D B.

```
def roll(s: Stack, n: int) -> None:
```

Question 3

There are many ways to implement ADTs. Implement the queue ADT below using a Python dictionary.

```
class Queue:

  '''A first-in, first-out (FIFO) queue of items'''

  def __init__(self: 'Queue') -> None:
    '''A new empty Queue.'''
    self._data = {}

  # implement enqueue, dequeue, is_empty, front
```