STACKS

Common Collections in Programming

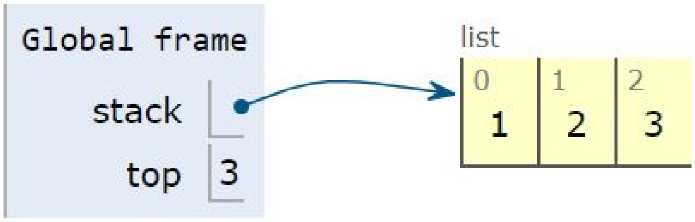
- Python lists are flexible
- can be used to implement other widely used data structures
  1. stacks
  2. queues

Stack

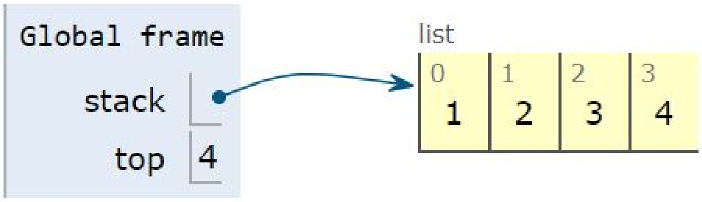
- a sequential collection
- Last-In-First-Out
- principal operations:
  1. push (add to the top)
  2. pop (remove from top)
  3. peek (examine top)
- built-in types in some languages

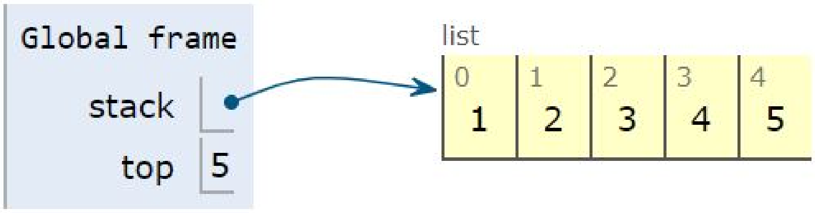
Stack Example

- stack contains 1,2,3



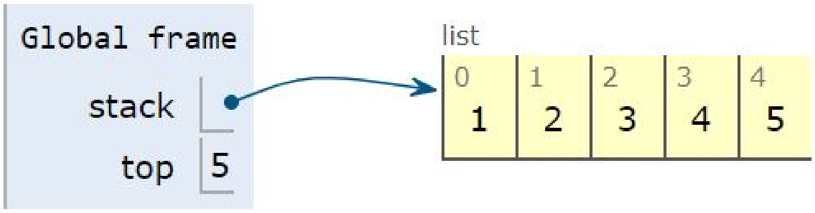
- push 4, then push 5



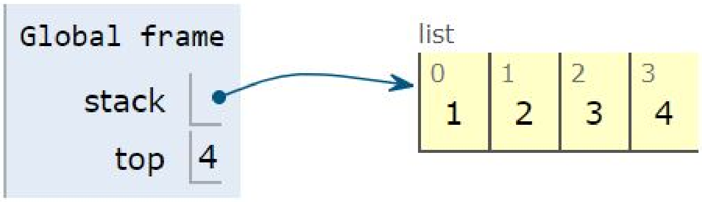


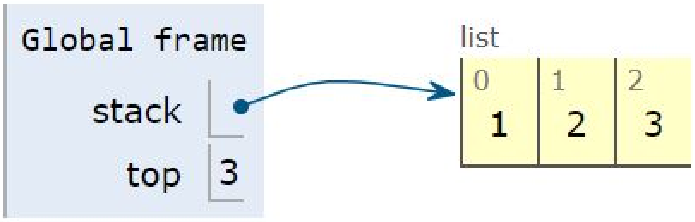
Stack Example (cont'd)

- peek (examine top)



- pop, then another pop





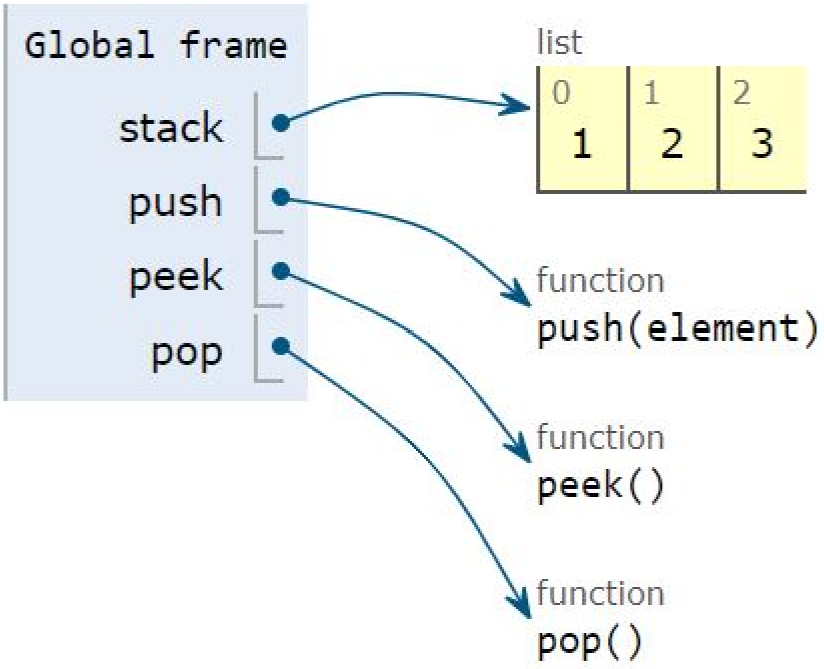
Stack with Python Lists

```
stack = []

def push(element):
    stack.append(element)

def peek():
    if len(stack) > 0:
        return stack[-1]
    else:
        return None

def pop():
    if len(stack) > 0:
        return stack.pop(-1)
    else:
        return None
```

- use list mutability

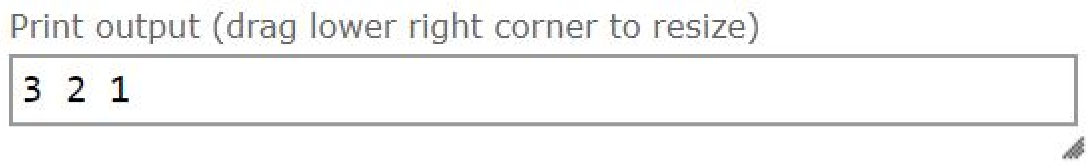
Stack w. Lists (cont'd)

```
push(1)
push(2)
push(3)
```



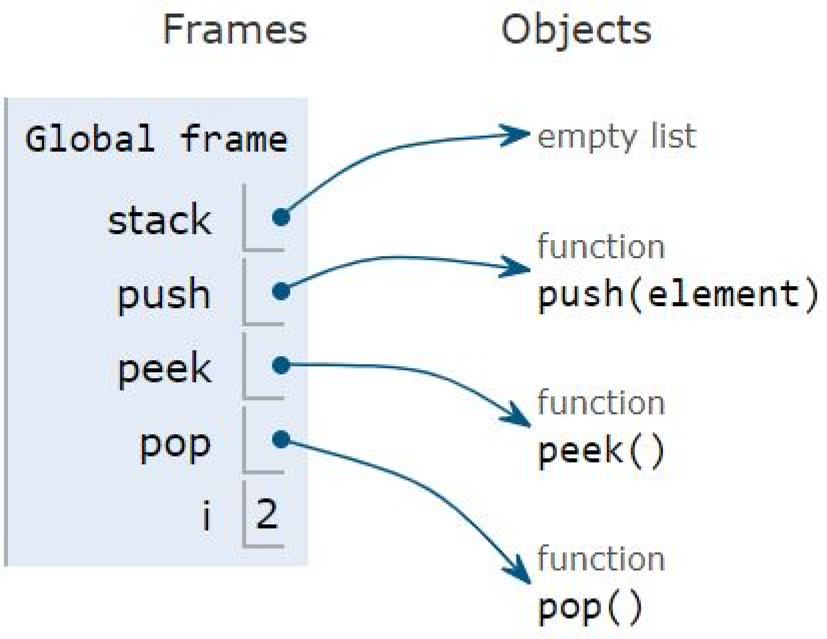
Stack w. Lists (cont'd)

```
for i in range(len(stack)):
    print(pop(), end=' ')
```



Print output (drag lower right corner to resize)

```
3 2 1
```

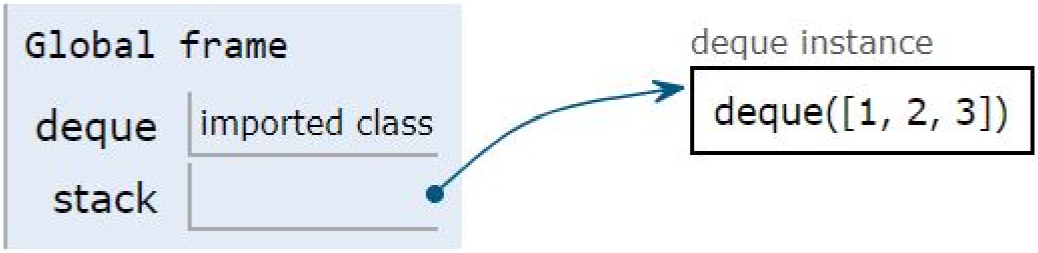


- Last-In-First-Out (LIFO)

Stack with Python Collections

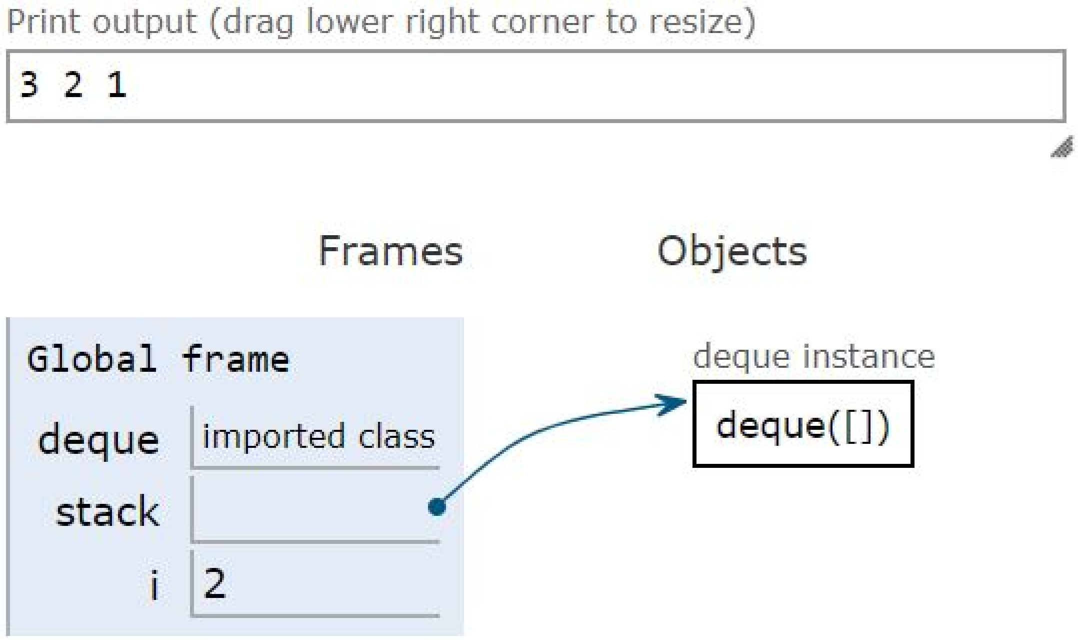
- use *deque* from *collections*
- faster for *push*() and *pop*()

```
from collections import deque

stack = deque()
stack.append(1)   # implements a push
stack.append(2)
stack.append(3)
```



Stack with Collections (cont’d)

```
for i in range(len(stack)):
    print(stack.pop(), end=’ ’)
```



Print output (drag lower right corner to resize)

```
3 2 1
```

Stack with Python LiFoQueue Object

- can use LiFoQueue object
- *put*() implements *push*()
- *get*() implements *pop*()

```
from queue import LifoQueue

stack  = LifoQueue(maxsize = 10)

stack.put(1)    # implements push
stack.put(2)
stack.put(3)

for i in range(stack.qsize()):
    print(stack.get(), end=' ')
```