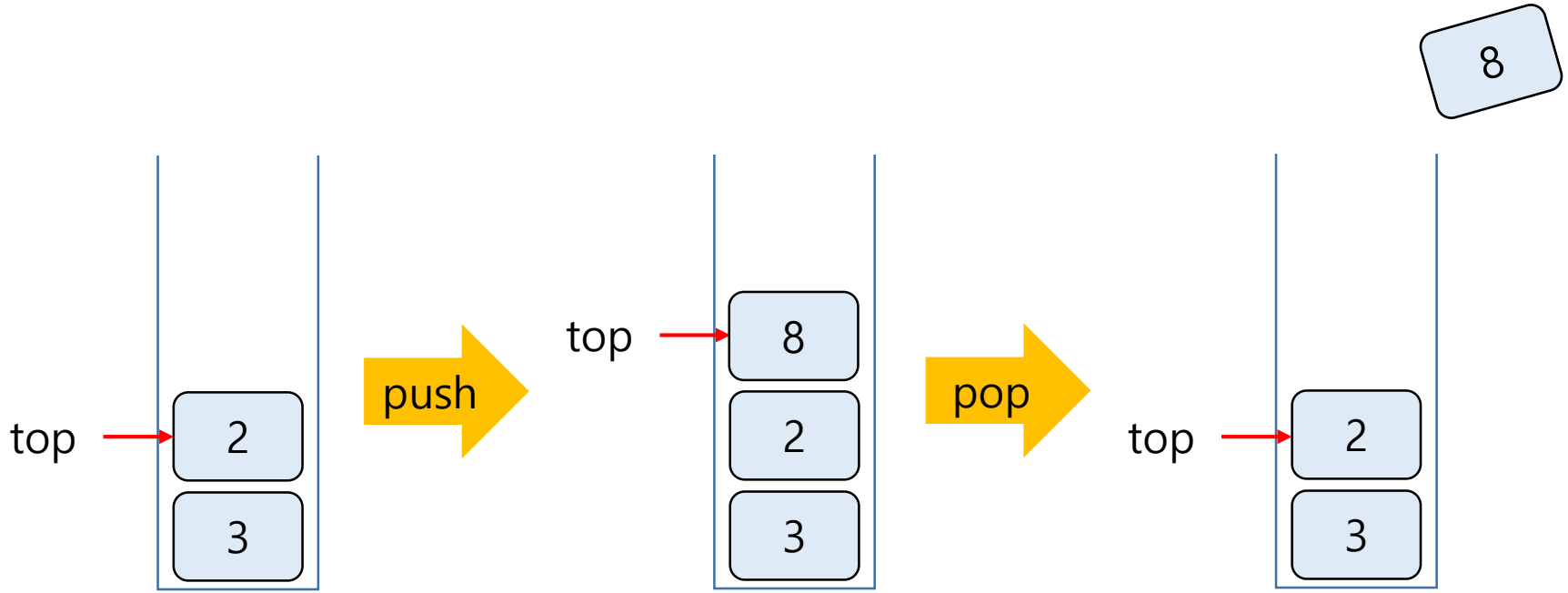
Stack (스택)

스택

- 특성
  - First In Last Out & Last In First Out
- 연산
  - push()
  - pop()



스택 구현

- 양방향 연결 리스트로 구현
  - 멤버변수 : 리스트
  - push() : 앞에 추가
  - pop() : 앞에서 제거

스택 구현 – 양방향 연결 리스트 활용

- 클래스 Stack 정의
  - one member variables
    - stack : 아이템들을 저장할 리스트. 객체 생성 시에 빈 리스트로 초기화
  - 생성자, push(), pop(), print_stack()

```
class Stack:
    def __init__(self):
        self.stack = Dlist.Dlist()
```

스택 구현 – 양방향 연결 리스트 활용

- 클래스 Stack 정의
  - push() & pop()

```
class Stack :
    ...
    def push(self, item):
        self.stack.insert_front(item)
    def pop(self):
        node = self.stack.delete_front()
        if (node == None):
            return None
        return node.item
```

스택 구현 – 양방향 연결 리스트 활용

- 클래스 Stack 정의
  - print_stack()

```
class Stack :
    ...
    def print_stack(self):
        self.stack.print_list()
```

스택 활용 프로그램!

```
s = Stack()

s.push('mango')
s.push('apple')
s.push('orange')
s.print_stack()

s.pop()
s.print_stack()
```

스택 활용 프로그램2

- 스택을 활용하여 괄호 일치 여부 검사하기
  - {{[()]}} : 일치
  - {{[]}) : 일치하지 않음
  - 알고리즘
    - 왼쪽 괄호 만나면 스택에 *push*
    - 오른쪽 괄호 만나면 스택에서 *pop*한 왼쪽 괄호와 매치하는지 비교
      - 매치하지 않으면 일치하지 않음.
    - 비교 끝난 후에도 여전히 스택에 왼쪽 괄호가 남아 있으면 일치하지 않음
    - 괄호 매치 정보를 *dictionary*를 이용하여 관리 (*key*: 왼쪽 괄호, *value*: 오른쪽 괄호)

스택 활용 프로그램2

```
s = Stack()
result = True

parenthesis_dict = {'{':'}', '[':']', '(':')'}
string = input("input a string : ")

left_parenthesis = parenthesis_dict.keys()
right_parenthesis = parenthesis_dict.values()
```

스택 활용 프로그램2

```
for char in string:
    if char in left_parenthesis :
        s.push(char)
    elif char in right_parenthesis :
        left = s.pop()
        if (left == None or parenthesis_dict[left] != char) :
            print("wrong match", left, char)
            result = False
            break
```

스택 활용 프로그램2

```
left = s.pop()
if (left != None):
    print("wrong match", left)
    result = False

if (result == True):
    print('right match')
```

Homework candidate

- 스택을 활용하여 회문 검사하기
  - 회문: 좌우 대칭 문자열
    - *aba, aa, a, aaccaa, …*

appendix

스택 구현

- 파이썬 리스트로 구현 (self-study)
  - 멤버변수 : 리스트
  - push() : 끝에 추가 (append())
  - pop() : 끝에서 제거 (pop())

스택 구현 – 파이썬 리스트 활용

- 클래스 Stack 정의
  - one member variables
    - stack : 아이템들을 저장할 리스트. 객체 생성 시에 빈 리스트로 초기화
    - 아이템을 삽입하고 삭제할 위치를 어떻게 파악 (top의 위치)?
  - 생성자, push(), pop(), print_stack()

```
class Stack:
    def __init__(self):
        self.stack = []
```

스택 구현 – 파이썬 리스트 활용

- 클래스 Stack 정의
  - push() & pop()

```
class Stack :
    ...
    def push(self, item):
        self.stack.append(item)

    def pop(self):
        if (len(self.stack) == 0):
            return None;
        return (self.stack.pop())
```

스택 구현 – 파이썬 리스트 활용

- 클래스 Stack 정의
  - print_stack()

```
class Stack :
    ...
    def print_stack(self):
        print(self.stack)
```

스택 활용 프로그램!

```
s = Stack()

s.push('mango')
s.push('apple')
s.push('orange')
s.print_stack()

s.pop()
s.print_stack()
```