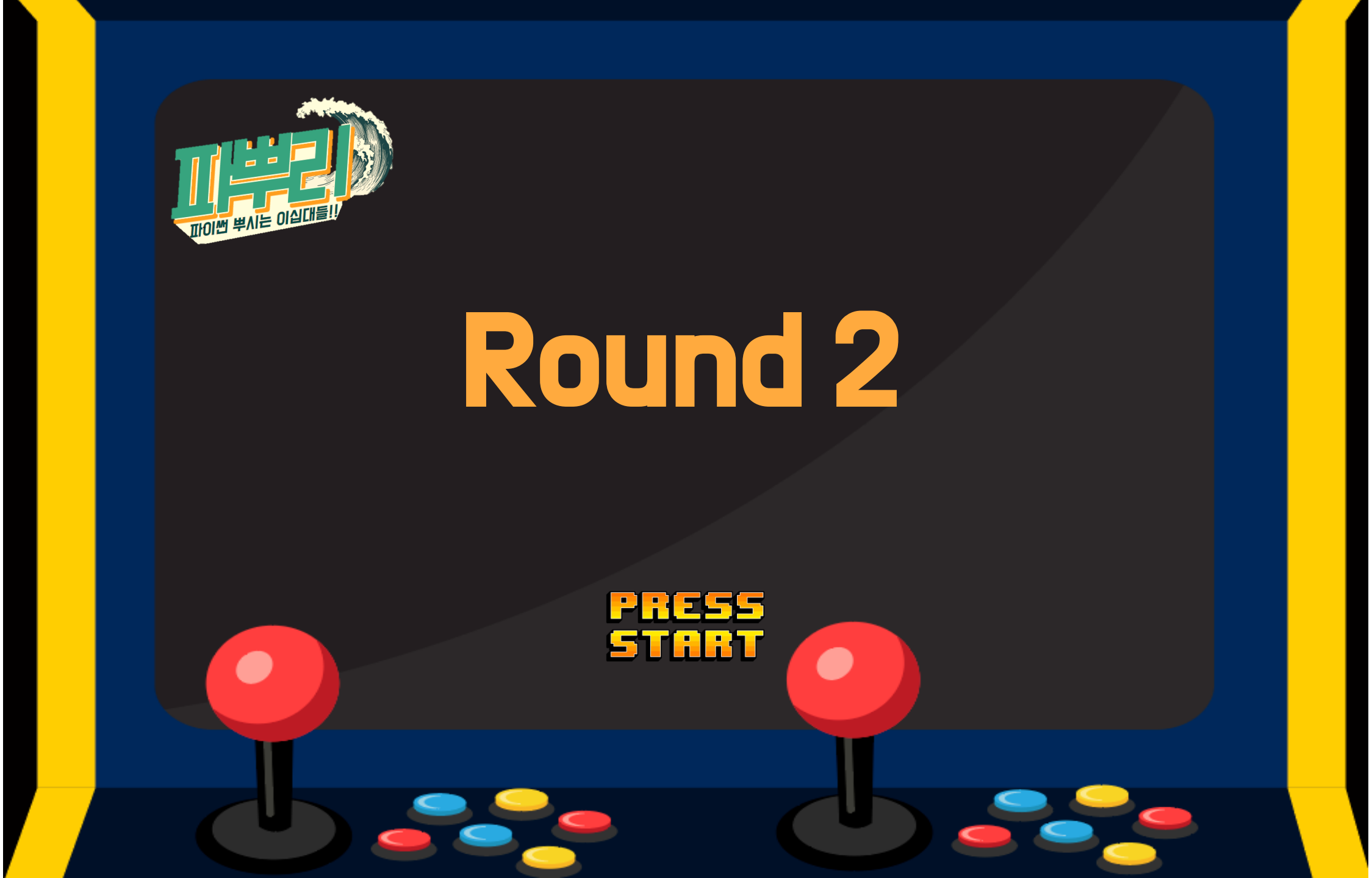

파뿌리
파이썬 뿌시는 이십대들!!
Round 2
PRESS
START

파뿌리
파이썬 뿌시는 이십대들!!

《 Round 2 》

- 모듈
- 클래스
- 예외처리
- 파이썬 자료구조

New Assignment

파뿌리
파이썬 뿌시는 이십대들!!

《 Round 2 》

- 모듈 《
- 클래스
- 예외처리
- 파이썬 자료구조

calculator.py

```
def add() :
def sub() :
def mul() :
def div() :
def rem() :

def result_write() :
def result_read() :

def start() :
    .
    .
```

cal_IO.py

def result_write() :

def result_read() :

cal_module.py

def add :

def sub :

def div :

def mul:

calculator.py

```
import cal_module
import cal_IO
import cal_start

start()
...
```

cal_start.py

def start() :

파뿌리
파이썬 뿌시는 이십대들!!

why ?

파뿌리
파이썬 뿌시는 이십대들!!

why ?

파뿌리
파이썬 뿌시는 이십대들!!

《 Round 2 》

- 모듈 - complete
- 클래스 《
- 예외처리
- 파이썬 자료구조

파뿌리
파이썬 뿌시는 이십대들!!

example Java class…

```
class Cal {
	private int result;

	public Cal(){
		result = 0;
	}

	public add(int num){
		result += num;
		system.out.println(result);
	}

	public sub(int num){
		result -= num;
		system.out.println(result);
	}
}
```

파뿌리
파이썬 뿌시는 이십대들!!

in Python class!

```
class Cal :
        def __init__(self) :
                self.__result = 0

        def add(self, num) :
                self.__result += num
                print(self.__result)

        def sub(self, num) :
                self.__result -= num
                pritn(self.__result)

...
```

파뿌리
파이썬 뿌시는 이십대들!!

in Python class!

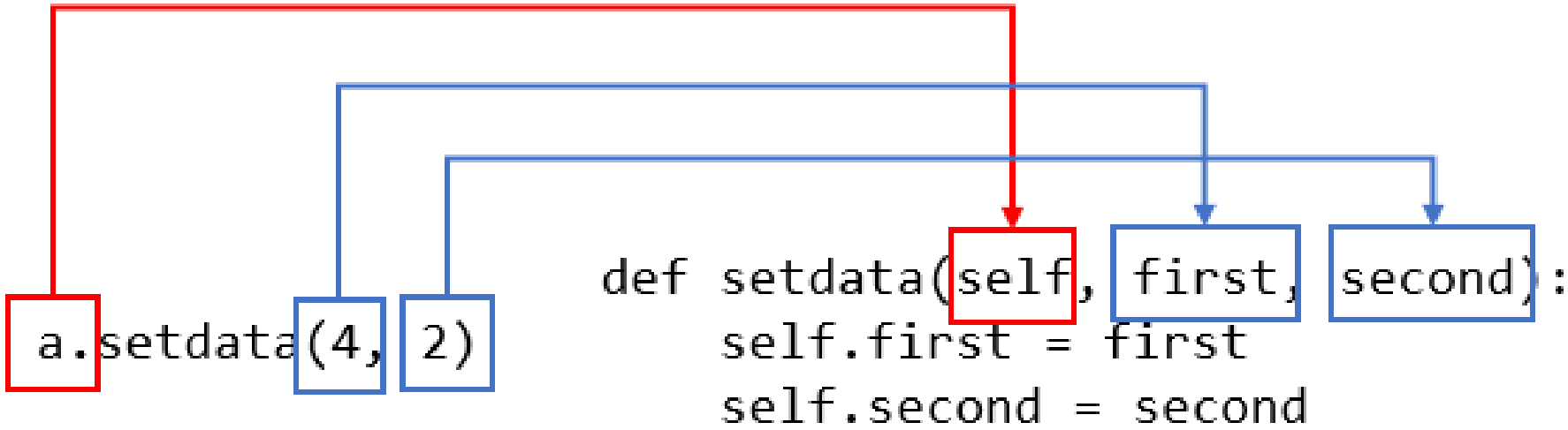
```
class Cal :
        def __init__(self) :
                self.__result = 0

        def add(self, num) :
                self.__result += num
                print(self.__result)

        def sub(self, num) :
                self.__result -= num
                pritn(self.__result)

...
```

파뿌리
파이썬 뿌시는 이십대들!!

python의 접근제한자는 작명을 통해 지정!

public	protected	private
접두사x ex) num	한 개의 밑줄이 접두사 ex) _num	두 개의 밑줄이 접두사 ex) __num
	접미사는 밑줄 한 개까지만 허용 ex) _num_ / __num_	
접미사에 밑줄 두 개 적용시 public으로 적용 ex) _num__ / __num__		

self?



2. 클래스 내의 함수는 자신의 호출을 명시적으로 써줘야 함.
(self라는 이름을 굳이 쓸 필요는 없지만. 관례적으로 self라고 이름을 써왔다고 합니다.)

파뿌리
파이썬 뿌시는 이십대들!!

__init__?

== 파이썬 클래스 생성자 o o
ex) def __init__(self)

3. 파이썬 클래스의 생성자는 __init__()이다.
(얘는 이름이 원래 __init__()임)

※ 클래스변수와 객체변수

(실수 많이 발생하는 부분)

```python
# 클래스변수 result와 객체변수 results
class Cal:
    result = 0
    def __init__(self):
        self.results = 0

cal1 = Cal()
cal2 = Cal()

# 클래스변수에 1 대입
Cal.result = 1
# 클래스의 객체변수에 1대입
Cal.results = 1
# cal2 객체변수에 1대입
cal2.results = 1

print(cal1.result, cal1.results)
print(cal2.result, cal2.results)
```

1)

```
class Cal :
    result = 0
    def __init__(self):
        self.results = 0

cal1 = Cal()
cal2 = Cal()
```

results = 0

인스턴스화(실체화)

cal1
cal1.result = 0
cal1.results = 0

cal2
cal2.result = 0
cal2.results = 0

2)

```
# 클래스변수에 1 대입
Cal.result = 1
```

해당 클래스로 만들어진
모든 인스턴스에 공유

cal1
cal1.result = 1
cal1.results = 0

cal2
cal2.result = 1
cal2.results = 0

3)

```
# 클래스의 객체변수에 1대입
Cal.results = 1
```

```
class Cal :
    result = 0
    def __init__(self):
        self.results = 0
```

results는 생성자를 통한 초기화를 했을 때만 생성됨.
즉, 해당 코드는 무용지물

4)

```
# cal2 객체변수에 1대입
cal2.results = 1
```

5)

```
print(cal1.result, cal1.results)
print(cal2.result, cal2.results)
```

cal1
cal2.result = 1
cal2.results = 0

cal2
cal2.result = 1
cal2.results = 1

```
1 0
1 1
```

파뿌리
파이썬 뿌시는 이십대들!!

《 Round 2 》

- 모듈 - complete
- 클래스 - complete
- 예외처리 《
- 파이썬 자료구조

파뿌리
파이썬 뿌시는 이십대들!!

example Java…

```
static Scanner sc = new Scanner(…);
int num1 ;
int num2 ;

try {
        num1 = sc.nextInt();
        num2 = sc.nextInt();
        System.out.println(num1/num2);

} catch (ArithmeticException e) {
        System.out.println("0으로 나누지 마!")

} catch (InputMismatchException e) {
        System.out.println("int형으로 넣어라")

} finally {
        System.out.println("try문이 끝났어.")
}
```

파뿌리
파이썬 뿌시는 이십대들!!

example Python…

```
try :
        num1 = int(input())
        num2 = int(input())
        print(num1/num2)

except ZeroDivisionError as e :
        print("0으로 나누지 마!")

except ValueError as e:
        print("int형으로 넣어라")

finally :
        print("try문이 끝났어.")
```

파뿌리
파이썬 뿌시는 이십대들!!

《 Round 2 》

- 모듈 - complete
- 클래스 - complete
- 예외처리 - complete
- 파이썬 자료구조 《

```python
# 리스트 자료형
ex_list = [1, "이", ,[3]]

# 튜블 자료형
ex_tuple = (1, "이", [3])

# 딕셔너리 자료형
ex_dictionary = {1 : 1,  2 : "이", 3 : [3]}

# 집합 자료형
ex_set = set([1, "이", [3]])
```

파뿌리
파이썬 뿌시는 이십대들!!

List

```
# 리스트 자료형
ex_list = [1, "이", ,[3]]
```

요소 수정	요소 형태	특징
O	Anything	대괄호

- array(배열)과 list(리스트)의 차이

배열은 크기가 정해져 있으며, 기능이 존재하지 않는다.
리스트는 크기가 가변적이며 여러 기능이 존재한다.

배열 인덱스는 값에 대한 유일무이한 식별자.
리스트 인덱스는 몇 번째 데이터인가 정도의 의미.

배열은 요소를 삭제해도 순서가 그대로이다.
리스트는 중간 요소를 삭제하면 순서가 바뀐다.

파뿌리
파이썬 뿌시는 이십대들!!

Tuple

```
# 튜블 자료형
ex_tuple = (1, "이", [3])
```

요소 수정	요소 형태	특징
X	Anything	소괄호

- 멀쩡한 리스트를 두고 튜플은 왜 쓰는거지??

들어 있는 요소의 수를 사전에 정확히 알고 있으면 튜플이 유리하다.

리스트에 비해 조금 더 공간 효율적이다.

…

그렇다고 한다…

파뿌리
파이썬 뿌시는 이십대들!!

Dictionary

```
# 딕셔너리 자료형
ex_dictionary = {1 : 1,  2 : "이", 3 : [3]}
```

요소 수정	요소 형태	특징
O	Key : Value(anything)	중괄호 index가 없음

- 자바와 C++의 Map형태와 유사하다.

- Dictionary자료형은 다양한 함수를 지원하기 때문에 알아두면 유용하다.
key의 리스트를 만들어주는 keys()
value의 리스트를 만들어주는 values()
해당 Dictionary의 특정 key를 탐색하는 in

- key값은 문자나 숫자, 튜플같은 변하지 않는 값만 들어감.

파뿌리
파이썬 뿌시는 이십대들!!

Set

```
# 집합 자료형
ex_set = set([1, "이", [3]])
```

요소 수정	요소 형태	특징
O	anything but set	set()을 통해 생성 중복 비허용 순서가 없음

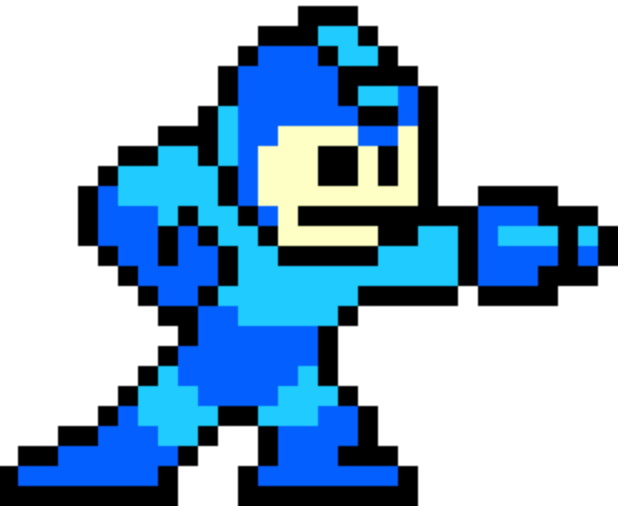
- Set은 어떤 경우에 쓴담?

자료의 중복을 제거할 때(많이 쓰지는 않음)

자료의 집합 연산을 할 때(합집합, 차집합, 교집합)
->> 집합연산 필요 시 가장 유용하게 쓸 수 있는 자료구조

파뿌리
파이썬 뿌시는 이십대들!!

WARNING



파뿌리
파이썬 뿌시는 이십대들!!

WARNING

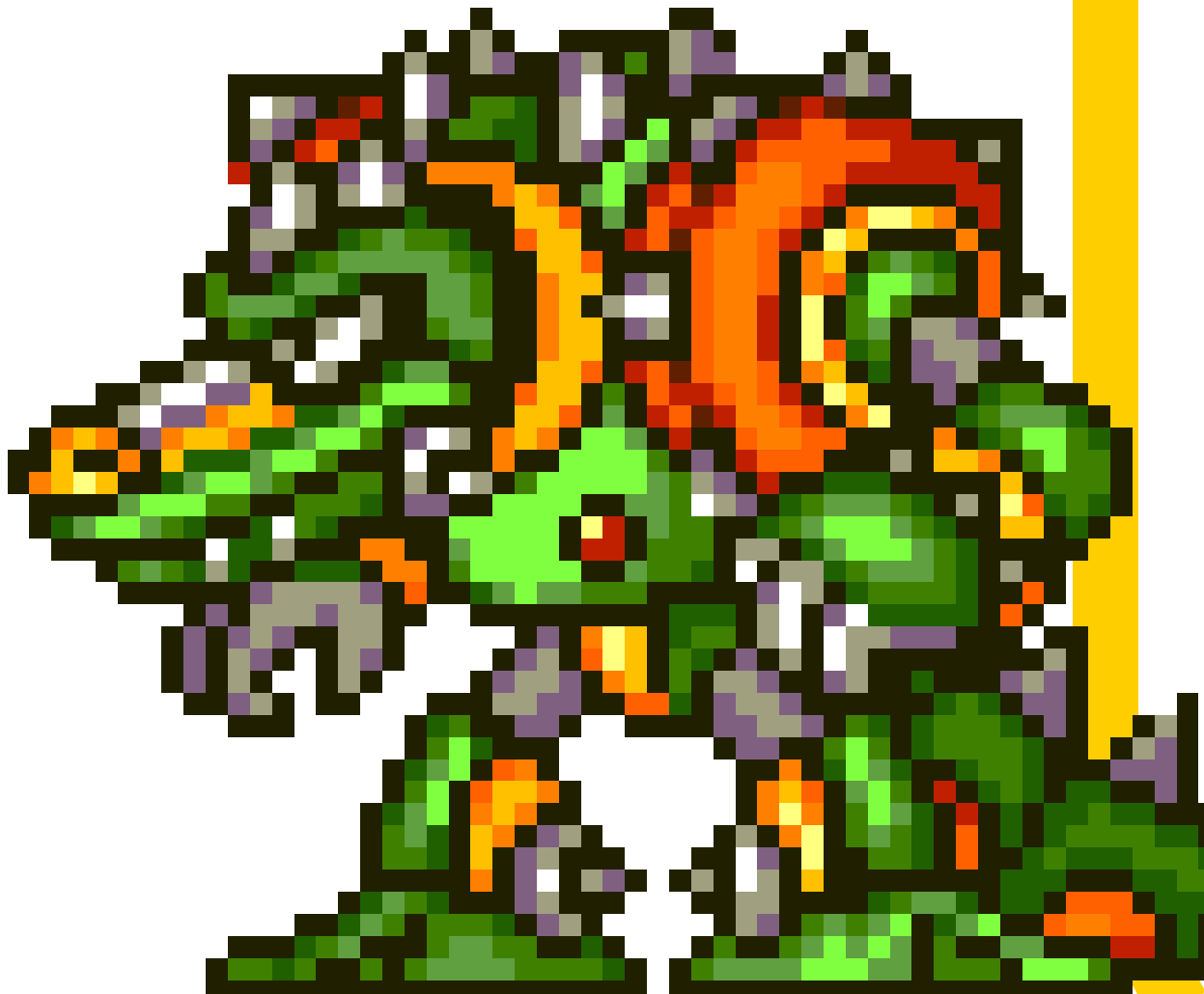
《 QUEST 》

- 각자 맡은 파트 발표자료 준비 -

Que/Stack/Deque/Linked-list

Graph/Tree(Binary, Heap Tree)

Sort Algorithm



파뿌리
파이썬 뿌시는 이십대들!!

NEXT STAGE