큐(Queue)

1. 큐 구조

- 줄을 서는 행위와 유사
- 가장 먼저 넣은 데이터를 가장 먼저 꺼낼 수 있는 구조
- 음식점에서 가장 먼저 줄을 선 사람이 제일 먼저 음식점에 입장하는 것과 동일
- **FIFO(First-In, First-Out)**또는 LILO(Last-In, Last-Out)방식으로 스택과 꺼내는 순서가 반대

2. 알아둘 용어

- Enqueue : 큐에 데이터를 넣는 기능
- Dequeue : 큐에서 데이터를 꺼내는 기능

3. 파이썬 queue라이브러리 활용해서 큐 자료 구조 사용하기

- queue라이브러리에는 다양한 큐 구조로 Queue(), LifoQueue(), PriorityQueue()제공
- 프로그램을 작성할 때 프로그램에 따라 적합한 자료 구조를 사용
  - Queue() : 가장 일반적인 큐 자료 구조
  - LifoQueue() : 나중에 입력된 데이터가 먼저 출력되는 구조(스택 구조라고 보면 됨)

- PriorityQueue() : 데이터마다 우선순위를 넣어서. 우선순위가 높은 순으로 데이터 출력

3.1 Queue()로 큐 만들기(가장 일반적인 큐, FIFO)

```
import queue

data_queue = queue.Queue()
```

```
data_queue.put("funcoding")
data_queue.put(1)
```

```
data_queue.qsize()

>>2
```

```
data_queue.get()
>>'funcoding'
```

```
data_queue.get()
>>1
```

```
data_queue.qsize()
>>0
```

3.2 PriorityQueue()로 큐 만들기

```
import queue

data_queue = queue.PriorityQueue()
```

```
data_queue.put((10, "korea"))#(우선순위,데이터)
data_queue.put((5, 1))
data_queue.put((15, "china"))
```

```
data_queue.qsize()
>>3
```

```
data_queue.get()
>>(5,1)
```

```
data_queue.get()
>>(10,'Korea')
```

참고 : 어디에 큐가 많이 쓰일까?

- **멀티 태스킹을 위한 프로세스 스케쥴링 방식**을 구현하기 위해 많이 사용됨(운영체제 참조)
- 너비 우선 탐색(BFS, Breadth-First Search)구현
  - 처리해야 할 노드의 리스트를 저장하는 용도로 큐 사용
  - 노드를 하나 처리할 때마다 해당 노드와 인접한 노드들을 큐에 다시 저장한다.
  - 노드를 접근한 순서대로 처리할 수 있다.
- 캐시(cache)구현

4. 프로그래밍 연습

4.1 연습1 : 리스트 변수로 큐를 다루는 enqueue, dequeue 기능 구현해 보기

```
queue_list= list()

def enqueue(data):
    queue_list.append(data)
def dequeue():
    data = queue_list[0]
    del queue_list[0]
    return data

for index in range(10):
    enqueue(index)
```

```
len(queue_list)
>>10
dequeue()
>>0
```

dequeue를 동작하면 0,1,2,3,...10순으로 빠져나온다.