FIFO_worker.py

```
from listQueue import ListQueue
import threading
import time
class Producer:
    def __init__(self, items, q1):
            self.q1 = q1
            self.__alive = True
            self.items = items
            self.pos = 0
            self.worker = threading.Thread(target=self.run)
    def get_item(self):
        if self.pos < len(self.items):
            item = self.items[self.pos]
            self.pos += 1
            return item
        else:
            return None
    def run(self):

        while True:
            time.sleep(0.2)
            if self.__alive:
                item = self.get_item()
                self.q1.enqueue(item)
                if item!=None:
                    print("Arrived:", item)
            else:
                break

        print("Producer is dying")
    def start(self):
        self.worker.start()
    def finish(self):
        self.__alive = False
        self.worker.join()
class Consumer:
    def __init__(self, q1):
        self.q1 = q1
        self.__alive = True
        self.worker = threading.Thread(target=self.run)
    def run(self):
        while not q1.isEmpty:
            pass
        while True:
            time.sleep(1)
            if self.__alive:
                if not self.q1.isEmpty():
                    print("Boarding:", self.q1.dequeue())
            else:
                break
        print("Consumer is dying")
    def start(self):
        self.worker.start()
    def finish(self):
        self.__alive = False
        self.worker.join()
if __name__ == "__main__":

    q1 = ListQueue()

    customers = []
    with open("customer.txt", 'r') as file:
```

```
        lines = file.readlines()
        for line in lines:
            customer = line.split()
            customers.append(customer)
    # FIFO
    names = []
    for c in customers:
        names.append(c[1])
    producer = Producer(names, q1)
    # Priority
    # producer = Producer(customers)
    consumer = Consumer(q1)
    producer.start()
    consumer.start()
    time.sleep(10)
    producer.finish()
    consumer.finish()
```

실행화면

```
user@WIN-L4EF0C26ORO:/mnt/c/Users/kimgu/Desktop/git/ds_2024/producer_consumer$ /bin/python3
/mnt/c/Users/kimgu/Desktop/git/ds_2024/producer_consumer/fifo_worker.py
Arrived: Alice
Arrived: Bob
Arrived: Charlie
Arrived: David
Boarding: Alice
Arrived: Eva
Arrived: Frank
Arrived: Grace
Arrived: Henry
Arrived: Ivy
Boarding: Bob
Arrived: Jack
Arrived: Kate
Arrived: Leo
Arrived: Mia
Arrived: Nick
Boarding: Charlie
Arrived: Olivia
Arrived: Peter
Arrived: Quinn
Arrived: Rachel
Arrived: Sam
Boarding: David
Arrived: Tina
Arrived: Ulysses
Arrived: Victoria
Arrived: Will
Arrived: Xavier
Boarding: Eva
Arrived: Yvonne
Arrived: Zachary
Arrived: Emily
Arrived: Ryan
Arrived: Sophia
Boarding: Frank
Boarding: Grace
Boarding: Henry
Boarding: Ivy
Boarding: Jack
Producer is dying
```

```
Consumer is dying
```

Prioty_worker.py

```
from listQueue import ListQueue
import threading
import time
class Producer:
    def __init__(self, items, norm, gold, plat):
            self.norm = norm
            self.gold = gold
            self.plat = plat
            self.__alive = True
            self.items = items
            self.pos = 0
            self.worker = threading.Thread(target=self.run)
    def get_item(self):
        if self.pos < len(self.items):
            item = self.items[self.pos]
            self.pos += 1
            return item
        else:
            return None
    def run(self):

        while True:
            time.sleep(0.2)
            if self.__alive:
                item = self.get_item()
                if item==None:
                    continue
                if item[0] == '1':
                    self.norm.enqueue(item[1])
                if item[0] == '2':
                    self.gold.enqueue(item[1])
                if item[0] == '3':
                    self.plat.enqueue(item[1])
                print("Arrived:", item[1])
            else:
                break

        print("Producer is dying")
    def start(self):
        self.worker.start()
    def finish(self):
        self.__alive = False
        self.worker.join()
class Consumer:
    def __init__(self, norm, gold, plat):
        self.norm = norm
        self.gold = gold
        self.plat = plat
        self.__alive = True
        self.worker = threading.Thread(target=self.run)
    def run(self):
        while True:
            time.sleep(1)
            if self.__alive:
                if not self.plat.isEmpty():
                    print("Boarding:", self.plat.dequeue())
```

```
                elif not self.gold.isEmpty():
                    print("Boarding:", self.gold.dequeue())
                elif not self.gold.isEmpty():
                    print("Boarding:", self.norm.dequeue())
            else:
                break
        print("Consumer is dying")
    def start(self):
        self.worker.start()
    def finish(self):
        self.__alive = False
        self.worker.join()
if __name__ == "__main__":

    norm = ListQueue()
    gold = ListQueue()
    plat = ListQueue()

    customers = []
    with open("customer.txt", 'r') as file:
        lines = file.readlines()
        for line in lines:
            customer = line.split()
            customers.append(customer)
    # Priority
    producer = Producer(customers, norm, gold, plat)
    consumer = Consumer(norm, gold, plat)
    producer.start()
    consumer.start()
    time.sleep(10)
    producer.finish()
    consumer.finish()
```

실행화면

```
user@WIN-L4EF0C26ORO:/mnt/c/Users/kimgu/Desktop/git/ds_2024/producer_consumer$ /bin/python3
/mnt/c/Users/kimgu/Desktop/git/ds_2024/producer_consumer/prioty_worker.py
Arrived: Alice
Arrived: Bob
Arrived: Charlie
Arrived: David
Boarding: Alice
Arrived: Eva
Arrived: Frank
Arrived: Grace
Arrived: Henry
Arrived: Ivy
Boarding: Eva
Arrived: Jack
Arrived: Kate
Arrived: Leo
Arrived: Mia
Arrived: Nick
Boarding: Frank
Arrived: Olivia
Arrived: Peter
Arrived: Quinn
Arrived: Rachel
Arrived: Sam
Boarding: Grace
Arrived: Tina
Arrived: Ulysses
```

```
Arrived: Victoria
Arrived: Will
Arrived: Xavier
Boarding: Mia
Arrived: Yvonne
Arrived: Zachary
Arrived: Emily
Arrived: Ryan
Arrived: Sophia
Boarding: Quinn
Boarding: Victoria
Boarding: Will
Boarding: Yvonne
Boarding: Ryan
Producer is dying
Consumer is dying
```