2021-2 자료구조및실습 실습과제 01

학번	2020136129	이름	최수연

1) 각 문제에 대한 분석과 및 해결 방법

1.1 Bag의 추상 자료형을 구현한 코드와 테스트 코드를 구현하고 테스트해 보라.

(1) 먼저 교재 21~22쪽의 코드를 구현하고, 다양한 입력으로 테스트하라. 22쪽의 테스트 코드를 수정하면 된다.
(2) 연습문제 1.2와 같이 numOf(bag, item) 연산을 추가하고, 이 연산이 잘 동작하는지 테스트하라.

[문제분석 및 해결방법]

(1) 교재 21쪽의 코드를 구현하고, myBag에 다양한 입력으로 테스트해 보았다. 먼저 'mybag'라는 새로운 빈 리스트를 생성한다. 첫 번째에는 bag.append(e) 연산을 사용하여 '스마트폰, 카드지갑, 립스틱, 집게핀, 에어팟, 충전기'를 myBag에 삽입하였고, print(myBag)를 통해 가방 속 물건 리스트 항목을 출력하도록 하였다. 또한 가방 속 물건의 개수를 len(bag)를 통해 계산하여 출력하도록 하였다. 이때 가방 속 물건의 개수는 6으로 출력된다. 그다음 두 번째는 bag.remove(e) 연산을 사용하여 myBag에 있는 '집게핀'을 삭제하고, 대신 bag.append(e) 연산으로 '머리끈, 립스틱'을 추가하였다. 마찬가지로 len(bag)를 통해 가방 속 물건의 개수를 출력한다. 이때 가방 속 물건의 개수는 7로 출력이 된다.

(2) 연습문제 1.2와 같이 numOf(bag, e) 함수 코드를 추가하여 가방 속의 특정 물건의 개수를 셀 수 있도록 하였다. bag.count(e) 연산을 사용하여 가방 속 립스틱의 개수를 세는 연산을 하였는데, 이때 첫 번째 mybag에서는 '스마트폰, 카드지갑, 립스틱, 집게핀, 에어팟, 충전기'가 들어있으므로, numOf(myBag, '립스틱')을 출력하였을 때 출력 결과는 1이 된다. 두 번째 mybag에서는 '집게핀'을 삭제하고, '머리끈, 립스틱'을 추가하였으므로, '스마트폰, 카드지갑, 립스틱, 에어팟, 충전기, 머리끈, 립스틱'이 된다. 이때 numOf(myBag, '립스틱')을 출력하면 출력 결과는 2가 된다. 이로써 mybag 속 립스틱은 1개에서 2개로 늘어난 것을 알 수 있다.

1.2 피보나치 수열과 관련된 다음 문제를 해결하라.

(1) 교재 37쪽, 38쪽과 같이 피보나치 수열을 순환적인 방법과 반복적인 방법으로 구하는 함수(fib(n), fib_iter(n))를 각각 작성하고, 결과가 정상적으로 출력되는지 테스트하시오.
(2) 1번째부터 39번째까지의 피보나치 값을 구해보자. 즉 두 함수의 입력 n으로 1~39를 전달하는 것이다. 각 함수의 처리 시간을 계산하여 출력하라. 이를 위해 24쪽의 코드와 설명을 참고하라. 이를 위한 반복문(for)은 다음과 같이 사용할 수 있다.
(3) 실행 결과를 분석해 보라. 두 함수의 시간 복잡도는 각각 얼마인지 설명하라.

[문제분석 및 해결방법]

(1) 교재 37쪽의 순환으로 구현한 피보나치수열 코드와 38쪽의 반복으로 구현한 피보나치수열 코드를 구현하고, print를 통해 순환과 반복 각각의 출력 코드를 구현하여 테스트해 보았다. 교재의 코드를 사용하여 순환과 반복으로 6번째 피보나치수열을 출력한 결과, 반복과 순환 모두 출력값이 5로 동일하게 출력되는 것을 알 수 있다. 피보나치수열은 0, 1, 1, 2, 3, 5 ... 로 진행되기 때문에 피보나치수열의 6번째는 5가 출력된다. (코드에 대한 설명은 **4)** 참고)

(2) 먼저 실행시간을 측정하기 위해 import time을 통해 time 모듈을 가져온다. 그리고 1~39번째까지의 피

보나치 값을 구하기 위해 반복 루프를 사용하여 범위를 1~39까지 진행되도록 설정한다. 이때 for in 문을 사용하려면 range는 1, 40으로 설정해야 1~39번째까지 계산이 된다. 시간 측정은 총 3번을 하는데, 순환과 반복 사이에 하나씩 t1, t2, t3 시간을 측정한다. t1과 t2 사이에는 반복으로 구현한 피보나치수열을 넣고, t2와 t3 사이에는 순환으로 구현한 피보나치수열을 넣는다. 그리고 반복과 순환 각각의 함수 처리 시간을 계산하여 출력한다. 출력 시 반복은 t2-t1, 순환은 t3-t2로 계산하여 출력한다.

(3) 실행 결과, 반복과 순환의 피보나치수열의 수열 순서와 수는 모두 동일하다는 것을 알 수 있다. 그러나 함수 처리 시간 측정 결과, 반복과 순환의 피보나치수열 함수 처리 시간은 다르다는 것을 알 수 있다. 반복으로 구현한 피보나치수열은 반복 루프를 사용하기 때문에 같은 항의 중복이 없어 각각의 처리 시간이 0.0초로 거의 동일한 것을 알 수 있다. 따라서 **반복적인 피보나치수열의 시간 복잡도는 O(n)**이다. 그러나 순환으로 구현한 피보나치수열의 경우 하나를 계산할 때, 그 밑의 항까지 모두 계산해야 하므로 같은 항이 계속 중복되어 계산된다. 예를 들어 fib(4)의 경우 fib(2)와 fib(3)을 순환 호출하는데, fib(2)는 fib(0)과 fib(1)이 필요하고, fib(3)은 fib(1)과 fib(2)를 순환 호출하며, 이때 fib(2)가 다시 한번 fib(0)과 fib(1)을 호출한다. 이처럼 n이 커질수록 함수 처리 시간이 더 오래 걸린다는 것을 알 수 있다. 따라서 **순환적인 피보나치수열의 순환 호출 과정을 계산해보았을 때, 시간 복잡도는 O(2^n)**라는 것을 알 수 있다.

2) 자신이 구현한 주요 코드

def numOf(bag, e) : return bag.count(e) ... print('가방속 물건 개수:', count(myBag)) print('가방속 립스틱의 개수:', numOf(myBag, '립스틱'))	numOf(bag, e) 함수로, bag 내 항목 e의 개수를 세는 함수이다. count메소드 bag.count(e)를 사용하였다. 미리 만들어 놓은 count(bag) 함수와 numOf(bag, e) 함수를 print를 사용하여 출력한 코드이다. count(bag)를 사용하여 가방 속 물건 개수, numOf(bag, e)를 사용하여 가방 속 립스틱의 개수를 출력하였다.

3) 다양한 입력에 대한 테스트 결과

1.1 Bag의 추상 자료형을 구현한 코드와 테스트 코드 구현하고 테스트해 보라.

```
수연이 가방속 물건: ['스마트폰', '카드지갑', '립스틱', '집게핀', '에어팟', '충전기']
가방속 물건 개수: 6
가방속 립스틱의 개수: 1
수연이 가방속 물건: ['스마트폰', '카드지갑', '립스틱', '에어팟', '충전기', '머리끈', '립스틱']
가방속 물건 개수: 7
가방속 립스틱의 개수: 2
```

1.2 피보나치 수열과 관련된 다음 문제를 해결하라.

```
Fibonacci반복(6번째) =  5
Fibonacci순환(6번째) =  5
n= 1    반복:  0.0 순환:  0.0
n= 2    반복:  0.0 순환:  0.0
n= 3    반복:  0.0 순환:  0.0
n= 4    반복:  0.0 순환:  0.0
n= 5    반복:  0.0 순환:  0.0
n= 6    반복:  0.0 순환:  0.0
n= 7    반복:  0.0 순환:  0.0
n= 8    반복:  0.0 순환:  0.0
n= 9    반복:  0.0 순환:  0.0
n= 10   반복:  0.0 순환:  0.0
n= 11   반복:  0.0 순환:  0.0
n= 12   반복:  0.0 순환:  0.0
n= 13   반복:  0.0 순환:  0.0
n= 14   반복:  0.0 순환:  0.0
n= 15   반복:  0.0 순환:  0.001146078109741211
n= 16   반복:  0.0 순환:  0.0005414485931396484
n= 17   반복:  0.0 순환:  0.0
n= 18   반복:  0.0 순환:  0.0010950565338134766
n= 19   반복:  0.0 순환:  0.0010023117065429688
n= 20   반복:  0.0 순환:  0.0020656585693359375
n= 21   반복:  0.0 순환:  0.002980947494506836
n= 22   반복:  0.0009448528289794922 순환:  0.0060040950775146484
n= 23   반복:  0.0 순환:  0.008996009826660156
n= 24   반복:  0.0 순환:  0.015211343765258789
n= 25   반복:  0.0 순환:  0.02693963050842285
n= 26   반복:  0.0 순환:  0.03937053680419922
n= 27   반복:  0.0 순환:  0.06800556182861328
n= 28   반복:  0.0 순환:  0.10859346389770508
n= 29   반복:  0.0 순환:  0.13770771026611328
n= 30   반복:  0.0 순환:  0.23287487030029297
n= 31   반복:  0.0 순환:  0.3894765377044678
n= 32   반복:  0.0 순환:  0.6191890239715576
n= 33   반복:  0.0 순환:  0.9856362342834473
n= 34   반복:  0.0 순환:  1.5721323490142822
n= 35   반복:  0.0 순환:  2.603374719619751
n= 36   반복:  0.0 순환:  4.1210479736328125
n= 37   반복:  0.0 순환:  6.78655743598938
n= 38   반복:  0.0 순환:  11.056476354598999
n= 39   반복:  0.0 순환:  18.92776083946228
```

4) 코드에 대한 설명 및 해당 문제에 대한 고찰

1.1.

contains(bag, e) 함수는 bag에 항목 e가 있는지 검사하는 함수로, in 연산자를 사용하여 T/F를 반환한다.
insert(bag, e) 함수는 bag에 항목 e를 삽입하는 함수로, append메소드 bag.append(e)를 사용한다.
remove(bag, e) 함수는 bag에 항목 e를 삭제하는 함수로, remove메소드 bag.remove(e)를 사용한다.
count(bag, e) 함수는 bag의 전체 항목 수를 계산하는 함수로, len(bag) 함수를 사용한다.
numOf(bag, e) 함수는 bag 내 항목 e의 개수를 계산하는 함수로, count메소드 bag.count(e)를 사용한다.
myBag = []를 통해 'myBag'이라는 새로운 리스트를 생성한다.

1.2

순환으로 구현한 피보나치수열의 경우, fib(n) 함수를 생성하여 n이 0일 때는 0, n이 1일 때는 1을 반환하고, 그 외에는 fib(n-1) + fib(n-2)를 반환하여 순환적으로 n이 0과 1이 나올 때까지 호출되도록 한다.

반복으로 구현한 피보나치수열의 경우, fib_iter(n) 함수를 생성하고, 먼저 n<2일 때에는 n의 값이 반환되도록 한다. 다음으로 last에 0, current에 1을 넣고, for in 문을 통해 range를 2, n+1로 설정하여 반복 루프를 만든다. for in 문 안에는 변수 tmp를 만들어 current의 값을 넣어두고, current에 last를 더한다. 그리고 current를 넣어둔 tmp를 이번에는 last로 옮기고 current를 반환한다. 이 과정을 range만큼 반복 실행하면 피보나치수열을 얻어낼 수 있다.

5) 이번 과제에 대한 느낀점

이번 강의와 과제를 통해 추상 자료형이라는 새로운 개념을 배우게 돼서 좋았다. 파이썬에 대해 잘 아는 것이 없어 첫 과제부터 낯설기도 했지만 C언어와는 다른 신선함을 느낄 수 있었다. 또한 복잡도 분석이나 빅오 표기법과 같은 수학적 개념도 파이썬 코드를 통해 배우면서 수의 연속성과 규칙을 잘 볼 수 있어서 좋았다. 특히 어떤 문제이냐에 따라 순환과 반복으로 구성된 각각의 코드의 시간 복잡도가 상반되는 결과를 보여주는 것이 재미있었다.

6) 궁금한 점이나 건의사항

딱히 없습니다.

7) 자신이 구현한 전체 코드

1.1

```
def contains(bag, e) :
    return e in bag

def insert(bag, e) :
    bag.append(e)

def remove(bag, e) :
    bag.remove(e)

def count(bag) :
    return len(bag)

def numOf(bag, e) :
    return bag.count(e)



myBag = []
insert(myBag, '스마트폰')
insert(myBag, '카드지갑')
insert(myBag, '립스틱')
insert(myBag, '집게핀')
insert(myBag, '에어팟')
insert(myBag, '충전기')
print('수연이 가방속 물건:', myBag)
print('가방속 물건 개수:', count(myBag))
print('가방속 립스틱의 개수:', numOf(myBag, '립스틱'))
```

```
remove(myBag, '집게핀')
insert(myBag, '머리끈')
insert(myBag, '립스틱')
print('수연이 가방속 물건:', myBag)
print('가방속 물건 개수:', count(myBag))
print('가방속 립스틱의 개수:', numOf(myBag, '립스틱'))
```

1.2

```
def fib(n) :
    if n == 0 : return 0
    elif n == 1 : return 1
    else :
        return fib(n - 1) + fib(n - 2)

def fib_iter(n) :
    if (n < 2): return n

    last = 0
    current = 1
    for i in range(2, n+1) :
            tmp = current
            current += last
            last = tmp
    return current

import time

print('Fibonacci반복(6번째) = ', fib_iter(5))
print('Fibonacci순환(6번째) = ', fib(5))
for i in range (1, 40) :
    t1 = time.time()
    fib_iter(i)
    t2 = time.time()
    fib(i)
    t3 = time.time()
    print("n=", i, "₩t반복: ", t2-t1, "순환: ", t3-t2)
```