자료구조 실습 보고서

[제 03주]

제출일:19.04.03

학번/이름: 201603867/조성환

1. 프로그램 설명서

1) 주요 알고리즘/ 자료구조/ 기타

LinkedBag은 4가지 기능을 담고 있다. 4가지는 add, remove, search, frequency인데, AppController에 의해 menu를 입력받고 menu에 해당되는 기능을 수행하며, 마지막에는 exit를 입력받고 끝난다.

LinkedBag은 Bag의 기능을 LinkedList로 구현한 자료구조이다. Bag이란 중복을 허용하며 원소들을 단순히 모아둔 자료구조이다. 이를 LinkedList로 구현을 하는 것이 바로 ArrayBag이다.

구현한 자료구조에는 원소 추가, 제거, 검색, 개수 조회가있다.

2) 함수 설명서

.ArrayBag생성자(int givenCapacity)

Arraybag객체를 생성할 때 최대 수용할 수 있는 개수를 입력받은 후 개수에 맞는 element를 set한 후 size를0으로 만든다.

.isEpmty()

Bag내 size가 0인지를 검사한다 맞으면 true를 반환한다.

.isFull()

false를 반환한다.

.doesContain(E anElement)

currentNode를 .head로 초기화 선언한다. while문을 통해 내용물을 참조한다. 만일 동일한 원소(.equals)가 존재한다면 true를 리턴한다. 만약 없으면 currentNode를 다음 Node로 초기화해준다.

마지막까지 없다면 (currentNode가 null일 때) false값을 리턴한다.

.frequencyOf(E anElement)

frequencyCount라는 int형 변수를 하나 0으로 초기화 선언한 후 currentNode를 _head로 초기화 선언을 한다. while문을 통해 내용물을 참조한다. 만일 동일한 원소(.equals)가 존재한다면 frequencyCount를 1 추가한다.이후 currentNode를 .next를 통해 다음 Node로 설정한다. 내용물을 모두 참조한 뒤 frequencyCount값을 리턴한다.

.add(E anElement)

if 문을 통해 Bag가 꽉 찼는지를 검사한다. 만일 꽉차있으면 false를 리턴한다.

else 문을 통해서 추가를 해준다. 새로운 노드LinkedNode의 객체를 생성한 후 생성한 노드의 element와 next를 설정한다. 이후 head를 newnode로 set한 후 size를 1 더해준다. 마지막으로 true를 리턴한다.

.remove(E anElement)

if를 통해 비어있는지를 확인한다. isEmpty()가 true를 반환하면 remove메소드는 false를 리턴한다.
else문에서 제거를 한다.
currentNode를 _head, previousNode를 null로 초기화 선언한 후, found라는 boolean형 변수를 false로 초기화 선언한다.
while문을 통해 내용물을 참조한다. 만일 동일한 원소(.equals)가 존재한다면 found를 true로 초기화해준다. 만일 찾으면 while문 조건에 !found가 있기 때문에 for문을 나오게된다.
만일 동일한 원소가 존재하지않는다면 currentNode를 .next를 통해 다음 Node로 바꿔준다.
이후 if문에서 !found가 true, 즉, found가 false일 경우 false값을 리턴해준다.
else문에서 previousNode의 .next를 currentNode.next로 지정한 후 size를 1줄여준다. 마지막으로 true를 리턴해준다.

3) 종합 설명서

ArrayBag은 Bag자료구조를 List로 구현한 형태이며, 기능은 Bag와 마찬가지로 원소들을 넣어두고 모아두는 역할을 한다. 구현한 기능으로는 add, remove,
mvc모델을 이용해 model, view, controller를 분리시켰다.
model에는 ArrayBag, Coin 클래스가있고
view에는 AppView 클래스가 있고
controller에는 AppController, main클래스가있다.

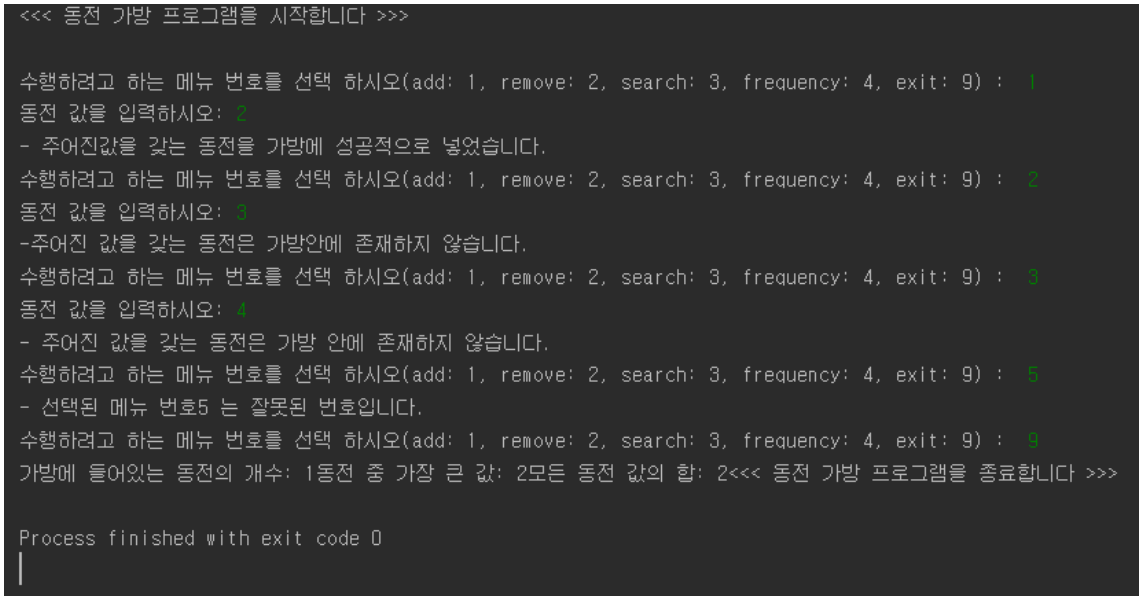
model.ArrayBag 클래스에서는 AppController.run에 의해 menunumber값을 입력받고 받은 입력에 따라 add, remove, frequency, elementAt의 메소드를 실행한다. 출력은 AppView에서 진행된다.

2. 프로그램 장단점/ 특이점 분석

예외처리를 하지 않았으며, 강의슬라이드에는 removeany, any 등 여러 메소드를 담고있으나 가장 간단한 4가지 기능만 구현을 해놓은 것이 단점이다.

3. 실행 결과 분석

1) 입력과 출력



2) 결과 분석

menu number 1234,9 그리고 잘못된 입력에 대해서는 제대로 처리를 하고 있으며, 잘못된 동전의 처리에 대해서도 잘 처리되어있다.

4. 소스 코드

.AppController.run

```
this.setCoinBag(new LinkedBag<Coin>());

int menuNumber= AppView.inputMenuNumber();
while (menuNumber!= MENU_END_OF_RUN){
    switch (menuNumber){
        case MENU_ADD:
            this.addCoin();
            break;

        case MENU_REMOVE:
            this.removeCoin();
            break;

        case MENU_SEARCH:
            this.searchForCoin();
            break;

        case MENU_FREQUENCY:
            this.frequencyOfCoin();
            break;

        default:
            this.undefinedMenuNumber(menuNumber);
            break;
    }
```

.LinkedBag

```java
//상태 알아보기
public boolean isEmpty() { return (this.size()== 0); }
public boolean isFull() { return false; }
public boolean doesContain(E anElement){
    LinkedNode<E> currentNode= this.head();
    while (currentNode!= null){
        if ((currentNode).element().equals(anElement)){
            return true;
        }
        currentNode= (currentNode).next();
    }
    return false;
}
```

```java
public boolean remove(E anElement){
    if (this.isEmpty()){
        return false;
    }else {
        LinkedNode<E> previousNode= null;
        LinkedNode<E> currentNode= this._head;
        boolean found= false;

        while (currentNode!= null && !found){
            if (currentNode.element().equals(anElement)){
                found= true;
            }else {
                previousNode= currentNode;
                currentNode= currentNode.next();
            }
        }

        if(!found){
            return false;
        }else {
            previousNode.set_next(currentNode.next());
        }
        this._size--;
        return true;

    }
}
```