**Argjira Shala** **1.assignment/10.task** 20th March 2022

QOYNNH

qoynnh@inf.elte.hu

Group 1

## Task

Implement the bag type which contains integers. Represent the bag as a sequence of (element, frequency) pairs. Implement as methods: inserting an element, removing an element, returning the frequency of an element, returning the largest element in the bag (suggestion: store the largest element and update it when the bag changes), printing the bag.

## Bag type

### Set of values

Bag={(element,frequency)* | element,frequency $\in$ Z}

### Operations

1. Insert an element into the bag

Inserting a given integer into the bag : B.add(e)

Formally: A:( B:Bag , e : Z)

Pre=(B=B')

Post=(B $\underline{u}$ {e})

2. Remove an element from the bag

Remove a given integer from the bag : B.removeElem(e)

Formally: A: (B : Bag , e : Z)

Pre= (B = B' $\wedge$ e $\in$ B)

Post = ($\forall$i $\in$ [1..|B|-1] : B.X[i].element = e -> B.removeElem(e)

3. Return the frequency of an element

Return the frequency of a given integer : B.get_Freq(e)

Formally: A: (B : Bag , e : Z , f : Z)

Pre= (B = B' $\wedge$ e $\in$ B)

Post = (Pre $\wedge$ $\forall$i $\in$ [1..|B|-1] : B.X[i].element = e -> f := Bag[e])

4. Return the largest element in the bag

Return the greatest integer inside the bag : B.largest()

Formally: A: (B : Bag , largest : Z )

Pre= (B = B' $\wedge$ |B| $\neq$ 0)

Post = (Pre $\wedge$ $\forall$i $\in$ [1..|B|-1]: B.X[i].element < B.X[i + 1].element : largest = B.X[|B|-1].element)

**Representation**

Ordered representation, binary search

Item = rec (element : Z , frequency : Z )

items : Item*

Inv : $\forall$i $\in$ [1.. |items|-1] : items[i].element < items [i + 1].element

**Implementation**

1. Insert an element into the bag

(exists, middlePoint):= logSearch(B, i.element)

If ¬ exists then B. items := B. items [1..middlePoint- 1] $\oplus$ <i> $\oplus$ B.items[middlePoint + 1 ..|items|] and i.frequency = 1

else B. items [middlePoint].frequncy = B. items [middlePoint].frequency + 1 endif

2. Remove an element from the bag

(exists, middlePoint):= logSearch(B, i.element)

If exists and B.items[middlePoint].frequency = 1 then B.items:= B.items[1... middlePoint - 1]⊕B. items[middlePoint + 1 ..|items|]

If exists and B.items[middlePoint].frequency > 1 then
B.items[middlePoint].frequncy = B.items[middlePoint].frequency – 1

else Error endif

3. Return the frequency of an element

(exists, middlePoint) := logSearch(B, i.element)

If l then f : = B.items[middlePoint].frequncy

else Error endif

4. Return the largest element in the bag

If |items| ≠ 0 then largest:=items[|items|-1.].element

else Error endif

5.logSearch algorithm

```
exists, upperbound , lowerbound := false , |items| , 1
while ¬exists and lowerbound ≤ upperbound loop
        middlePoint:=[(lb + ub)/2]
        if element < items[middlePoint].element then
            upperbound := middlePoint – 1
        else if element > items[middlePoint].element then
             lowerbound := middlePoint +1
        else
              exists : = true
        endif
end loop
if ¬exists then middlePoint:= lowerbound
```

# Testing

## Testing the operations (black box testing)

1. Creating the bag and checking the emptiness

a) The just created bag is empty.

b) Add an element and the bag will not be empty.

c) Remove the just entered element and the bag will be empty again.

2. Adding different elements once

a) Create the bag

b) add an element and check if the bag contains only this element, then check if the item's element in the bag is this element and its frequency of the item is 1 since it has been added only once.

c) add other elements , check if the bag contains these elements, check if their frequency is 1.

3. Adding different elements twice

a) Create the bag

b) add an element twice and check if the bag contains only this element, then check if the item's element in the bag is this element and its frequency of the item is 2 since it has been added twice.

c) add other elements twice , check if the bag contains these elements, check if their frequency is 2.

4. Add the same element more than two times

a) Create the bag

b) add an element more than two times and check if the bag contains only this element, then check if the item's element in the bag is this element and its frequency of the item is as many times as you have added it .

6. Add different elements different times (so they will have different frequency)

a) Create the bag

b) add different  elements ,different times and check if the bag contains all of them, then check if the item's elements in the bag are these elements and their frequency of the is as many times as you have added them .

7. Add different elements only once and remove them

a) Create the bag

b) add an element and check if the bag contains only this element, then check if the item's element in the bag is this element and its frequency of the item is 1 since it has been added only once. Then remove this element and check that there is nothing in the bag.

c) add other elements only once , check if the bag contains these elements, check if their frequency is 1. Begin to remove these elements and check how the size of the bag changes.

8. Add different elements three times(or more) and remove them only once

a) Create the bag

b) add an element three times and check if the bag contains only this element, then check if the item's element in the bag is this element and its frequency of the item is 3 since it has been added three times. Then remove this element only once and check that this element will still be in the bag but with frequency 2.

c) add other elements three times , check if the bag contains these elements, check if their frequency is 3.Begin to remove these elements only once and check how their frequency changes to 2 but the size of the bag does not change because these elements will be still in the bag.

9. Add different elements, different times and remove them different times

a) Create the bag

b)add an element different times, check if the bag has only this element, check that the frequency of this element is as many times as it was added . Remove this element different times and check how the frequency changes.

c) add other elements in the bag, different times(frequency), remove these elements different times, check how their frequency changes and check how the size of the bag may or may not change.

10. Get the largest element

a) Create the bag then add one element, this element should be the largest, add another element greater than the first one , now the newly added element will be the largest, add another one greater than the two previous ones and now this is the largest element. Add them different times and remove them and check how the largest element changes.

11. Get the frequency of an element

a) Create the bag, add different elements, different times, check how their frequency changes when adding and removing them.

**Testing based on code (white box testing)**

1. Check if removing an element throws an error

2. Check if getting the largest element throws an error

3. Check if getting the frequency of an element throws an error