Laboratory 1 - Creating set data structure implementation

Pijus Petkevičius

April 25, 2023

Contents

1. Task

Implement the set type which contains integers. Represent the set as a sequence of its elements. Implement as methods: inserting an element, removing an element, returning whether the set is empty, returning whether the set contains an element, returning a random element without removing it from the set, returning the number of even numbers in the set (suggestion: store the number of even numbers and update it when the set changes), printing the set. A set can store every element only once.

2. Set

2.1. Set of values

$Set(n) = \{a \in \mathbb{Z} \mid \forall i, j \in [1..n] : i \neq j \to a[i] \neq a[j]\}$

2.2. Operations

1. Insert an element (changes even elementCount)

   $A = (s : Set, e : \mathbb{Z})$
   $Pre = (s = s')$
   $Post = (Pre = s = s' \cup \{e\})$
   $s =$ output values, $s' =$ input values

2. Remove an element (changes even element count)

   $A = (s : Set, e : \mathbb{Z})$
   $Pre = (s = s' \wedge |s| > 0)$
   $Post = (Pre \wedge s = s' \backslash \{e\})$
   $s =$ output values, $s' =$ input values

3. Check if set is empty

   $A = (s : Set, l : \mathbb{L})$
   $Pre = (s = s')$
   $Post = (Pre \wedge l = (|s'| = 0))$

4. Check if set contains number

   $A = (s : Set, e : \mathbb{Z}, l : \mathbb{L}, ind : \mathbb{Z})$
   $Pre = (s = s' \wedge e = e')$
   $Post = (Pre \wedge (l, ind) = SEARCH_{i=0..|s.values|}(s.values[i] = e))$

5. Return random element

   $A = (s : Set, r : \mathbb{Z})$
   $Pre = (s = s')$
   $Post = (Pre \wedge r = s.values[rand()])$

6. Return the number of even elements in the section

   $A = (s : Set, r : \mathbb{N})$
   $Pre = (s = s')$
   $Post = (Pre \wedge r = s.evenElements)$

2.3. Representation

To store the set, we can use:

$$s = \{s_1, s_2..., s_n\} \qquad \leftrightarrow \qquad values =< v_1, v_2, ..., v_n >$$

To store the number of even numbers in the set, we will use:

$$evenElements : \mathbb{N}$$

2.4. Implementation

1. Insert an element
   Inserting Element to the set should check whether the element already exists in the set. If it does not exist, then we add element to the back of the vector and increase even elements count if element is even.

```
IF ¬s.containsElement(e)
    s.values:= s.values.Add(e)
    IF 2|e
        s.evenElements:= s.evenElements+1
    ELSE
        ∅
ELSE
    ∅
```

2. Remove an element
   Removing element from the set should check whether element exists. If yes, we find the position of the element. If element is even, we decrease evenElement by 1. then we assign index element the last element value and pop back the last element.

```
IF s.containsElement(e)
    l,i:=false,1
    WHILE $\neg l \wedge i \leq |s.values|$
        l,ind:=(s.values[i]=e),i
        i:=i+1
    IF 2|e
        s.evenElements:= s.evenElements-1
    ELSE
        ∅
    s.values[ind]:=s.values[|s.values|]
    s.values.RemoveAt(|s.values|)
ELSE
    ∅
```

3. Check if set is empty
   We should check if the length of the list is 0.

```
l:= false
l := |s.values|=0
```

4. Check if set contains number
   We Search through the list with condition $s.values[i] = e$. If element is found, l is assigned to true.

```
l,i:=false,1
WHILE $\neg l \wedge i \leq |v.values|$
    l,ind:=(s.values[i]=e),i
    i:=i+1
```

5. Return random element

```
r:=s.values[rand()]
```

6. Return the number of even elements in the section

```
r:=s.evenElements
```

3. Testing

3.1. Black box testing

1. insert method
    (a) Insert a single element into an empty set. (should be one element)
    (b) Insert multiple unique elements into a set. (should be all unique elements)
    (c) Insert a duplicate element into an empty set. (should be only 1 element)
2. remove method
    (a) Remove a single element from a set. (an element should be removed)
    (b) Remove a non-existent element from a set. (set should not change)
3. CheckIfEmpty method
    (a) Check if an empty set is empty.
    (b) Check if a nonempty set is empty.
4. containsElement method
    (a) Check if a set contains a single element.
    (b) Check if a set does not contain an element.
5. getRandomElement method
    (a) Get a random element from an empty set.
    (b) Get a random element from a nonempty set.
6. countEvenNumbers method
    (a) Count the number of even numbers in an empty set.
    (b) Count the number of even numbers in a set with no even numbers.
    (c) Count the number of even numbers in a set with even numbers.

3.2. White box testing

1. test size 0,10000