***Documentation***

16. ***ADT Bag – implementation on a hash table,***

***collision resolution by coalesced chaining***

Name: *Todoran Ana-Corina*

Group: *917*

Contents

[*1)* *ADT Specification* 1](#_Toc482728744)

[*2)* *ADT Interface* 2](#_Toc482728745)

[*a.* *Bag* 3](#_Toc482728746)

[*b.* *Iteraror* 4](#_Toc482728747)

[*3)* *ADT Representation* 5](#_Toc482728748)

[a. *Bag (Implemented on a hash table, collision resolution by coalesced chaining)* 5](#_Toc482728749)

[*b.* *Iterator* 5](#_Toc482728750)

[*4)* *Problem Statement* 5](#_Toc482728751)

1. *ADT Specification:*

* **Bag** = { b | b – bag with elements of type TElement }
* **TElement** -> the general element in containers

The interface of TElem contains the following operations:

* assignment (e1 ← e2)
  + - **pre**: e1, e2 ∈ TElem
    - **post**: e′1 = e2
* equality test (e1 = e2)
  + - **pre**: e1, e2 ∈ TElem
    - **post**:

True, if e1 = e2

equal =

False, otherwise

* **Iterator =** { it | it – iterator over Bag }

1. *ADT Interface:*
   1. *Bag****:***

* ***init( ):***
  + **pre**: True
  + **post**: b Bag
  + **throws**: - (None)
* ***destroy( b ):***
  + **pre**: b Bag
  + **post**: b is destroyed
  + **throws**: - (None)
* ***add( b, e )****:*
  + **pre**: b Bag **,** e TElement
  + **post**: b′ Bag, b′ = b + { e }
  + **throws**: - (None)
* ***remove( b, e )****:*
  + **pre**: b Bag **,** e TElement
  + **post**: b′ Bag, b′ = b - { e }
  + **throws**: - (None)
* ***size( b )****:*
  + **pre**: b Bag
  + **post**: size = the number of elements in b
  + **throws**: - (None)
* ***search( b, e ):***
  + **pre**: b Bag **,** e TElement
  + **post**:

True, if e is in b

search =

False, otherwise

* + **throws**: - (None)
* ***resize( b )****:*
  + **pre**: b Bag
  + **post**: b′ Bag, rehash( b )
  + **throws**: - (None)
* ***rehash( b )****:*
  + **pre**: b Bag
  + **post**: b′ - bag, m′ = 2 \* m
  + **throws**: - (None)
* ***iterator( b, it ):***
  + **pre**: b Bag
  + **post**: it Iterator, it – iterator over b
  + **throws**: - (None)
  1. *Iteraror****:***
     + ***init( b ):***
       - **pre**: b Bag
       - **post**: it Iterator , it – iterator over b pointing to ”first element”
       - **throws**: - (None)
     + ***next( it ):***
       - **pre**: it Iterator, it is a valid iterator
       - **post**: it′ - pointing to the next element
       - **throws**: - (None)
     + ***valid( it ):***
       - **pre**: it Iterator
       - **post**:

True, if it valid

valid(it) =

False, otherwise

* + - * **throws**: - (None)
    - ***getCurrent( it, e ):***
      * **pre**: it Iterator
      * **post**: e TElement, e – the current element pointed by it
      * **throws**: - (None)
    - ***begin( it, b ):***
      * **pre**: b Bag
      * **post**: it Iterator , it – iterator over b pointing to first element
      * **throws**: - (None)
    - ***end( it, b ):***
      * **pre**: b Bag
      * **post**: it Iterator , it – iterator over b pointing to last element
      * **throws**: - (None)

1. *ADT Representation:*
   1. *Bag (Implemented on a hash table, collision resolution by coalesced chaining)*:
      * length : Integer
      * m (capacity) : Integer
      * T : TElement[]
      * next : Integer[]
      * firstFree : Integer
      * h : Tfunction
   2. *Iterator*:
      * b : ↑Bag
      * currentPos : ↑Node
2. *Problem Statement:*

Having a password database, memorised by it’s password code, check if a given password exists. Password format consisting in numbers ( each number has to have at least 4 digits and maximum 10 ). A password is not necessarly unique, so for the given password if it exists show also the number of apparitions.