Documentație

(**TAD Colecție**, implementare folosind o listă dublu înlănțuită cu înlănțuirile reprezentate pe tablou)

**Nume și prenume:** Mihalache Florin

**Grupa:** 214

# Specificare și interfață TAD

**Colectie** = {**col** | **col** este o colecţie cu elemente de tip TElement}

***creează*(col)**

**pre:** -

**post:** col∈**Colectie**, col este colecţia vidă (fără elemente)

***adaugă*(col, e)**

**pre:** col ∈ **Colectie**, e ∈ TElement

**post:** col’ ∈ **Colectie**, col’= col ∪{e}

{s-a adăugat elementul în colecţie}

***şterge*(col, e)**

**pre:** col ∈ **Colectie**, e∈TElement

**post:** col’∈ **Colectie**, col’= col - {e}

{s-a eliminat o apariţie a elemetului din colecţie}

***caută*(col, e)**

**pre:** col ∈ **Colectie**, e ∈ TElement

**post:** ***caută*** = adevărat dacă e ∈ col

fals în caz contrar

***dimensiune*(col)**

**pre:** col ∈ **Colectie**

**post:** ***dimensiune*** = dimensiunea colecţiei (numărul de elemente) ∈ *N*\*

***vidă*(col)**

**pre:** col ∈ **Colectie**

**post:** ***vidă*** = adevărat în cazul în care colecţia vidă

fals în caz contrar

***iterator*(col, i)**

**pre:** col ∈ **Colectie**

**post:** i∈I, i este un iterator pe colecţia col

***distruge*(col)**

**pre:** col ∈ **Colectie**

**post:** colecţia col a fost 'distrusă' (spaţiul de memorie alocat a fost eliberat)

**IteratorColectie** = {**i** | **i** este un iterator pe o colecție cu elemente de tip TElement}

***creeaza*(i, c)**

**pre:** c ∈ **Colectie**

**post:** i ∈ **IteratorColectie,** s-a creat iteratorul i pe colecția c (elementul

*curent* din iterator referă ‘primul’ element din colecție)

***element*(i, e)**

**pre:** i ∈ **IteratorColectie**

**post:** e ∈ TElement, e este elementul curent din iterație

***valid*(i)**

**pre:** i ∈ **IteratorColectie**

**post:** ***valid*** = adevărat, dacă *curent* referă o poziție validă din colecție

fals, altfel

***următor*(i)**

**pre:** i ∈ **IteratorColectie**

**post:** *curent*’ referă următorul element din colecție

***anterior*(i)**

**pre:** i ∈ **IteratorColectie**

**post:** *curent*’ referă elementul anterior din colecție

# Proiectare și implementare TAD

# folosind o listă dublu înlănțuită cu înlănțuirile reprezentate pe tablou

**Nod**

element: ***TElement*** {elementul de tip ***TElement***}

urmator: ***Intreg*** {poziția din vector a nodului următor}

anterior: ***Intreg*** {poziția din vector a nodului anterior}

**Colectie**

colectie: ***Nod[]*** {vector cu elementele din colectie, stocate in ***Nod***}

prim: ***Intreg*** {poziția primului element}

primLiber: ***Intreg*** {prima poziție liberă din vector}

ultim: ***Intreg*** {poziția ultimului element}

dim: ***Intreg*** {numărul elementelor din colecție}

**IteratorColectie**

colectie: ***Colectie*** {colecția iterată}

curent: ***Intreg*** {poziția elementului curent din iterație}

Colectie

**subalgoritmul** *creează*(col) **este {Ɵ(n)}**

col.prim ← -1

col.primLiber ← 0

col.ultim ← -1

col.dim ← 0

**pentru** i ← 0, CAPACITATE\_MAXIMA-1 **execută**

*{Pregătesc spațiile libere, CAPACITATE\_MAXIMA este o constanta}*

col.colectie[i].anterior ← i - 1

col.colectie[i].urmator ← i + 1

**sfPentru**

col.colectie[0].anterior ← -1

**sfCreează**

**subalgoritmul** *adaugă*(col, e) **este {Ɵ(1)}**

poz ← col.primLiber *{poz:* ***Intreg****}*

col .primLiber ← col.colectie[poz].urmator

col.colectie[poz].element ← e

col.colectie[poz].urmator ← -1

col.colectie[poz].anterior ← col.ultim

**dacă** col.dim ≠ 0 **atunci**

col.colectie[ultim].urmator ← poz

**sfDacă**

col.ultim ← poz

**dacă** col.prim = -1 **atunci**

col.prim ← poz

**sfDacă**

col.dim ← col.dim + 1

**sfAdaugă**

**subalgoritmul** *șterge*(col, e) **este {O(n)}**

*{i:* ***Intreg****, gasit:* ***Bool****}*

i ← col.prim

gasit ← fals

**câtTimp** (i ≠ -1) **și** (¬gasit) **execută**

**dacă** col.colectie[i].element = e **atunci**

gasit ← true

**altfel**

i ← col.colectie[i].urmator

**sfDacă**

**sfCâtTimp**

**dacă** i = col.prim **atunci** *{****e*** *se află pe prima poziție}*

col.prim ← col.colectie[i].urmator

col.colectie[i].anterior ← -1

col.colectie[i].urmator ← col.primLiber

col.colectie[i].anterior ← -1

col.primLiber ← i

**altfel**

**dacă** i = col.ultim **atunci** *{****e*** *se află pe ultima poziție}*

col.ultim ← col.colectie[i].anterior

col.colectie[ultim].urmator ← -1

col.colectie[i].urmator ← col.primLiber

col.colectie[i].anterior ← -1

col.primLiber ← i

**altfel** *{****e*** *se află în interior}*

col.colectie[col.colectie[i].anterior].urmator ←

col.colectie[i].urmator

col.colectie[col.colectie[i].urmator].anterior ←

col.colectie[i].anterior

col.colectie[i].urmator ← col.primLiber

col.colectie[i].anterior ← -1

col.primLiber ← i

**sfDacă**

**sfDacă**

col.dim ← col.dim – 1

**sfȘterge**

**funcția** *dimensiune*(col) **este {Ɵ(1)}**

***dimensiune***← col.dim

**sfDimensiune**

**funcția** *vidă*(col) **este {Ɵ(1)}**

***vidă***← col.dim = 0

**sfVidă**

**subalgoritmul** *iterator*(col, i) **este {Ɵ(1)}**

creează(i, col)

**sfIterator**

**funcția** *caută*(col, e) **este {O(n)}**

*{i:* ***Intreg****, gasit:* ***Bool****}*

i ← col.prim

gasit ← fals

**câtTimp** (i ≠ -1) **și** (¬gasit) **execută**

**dacă** col.colectie[i].element = e **atunci**

gasit ← true

**altfel**

i ← col.colectie[i].urmator

**sfDacă**

**sfCâtTimp**

***caută*** ← gasit

**sfCaută**

IteratorColectie

**subalgoritmul** *creează*(i, c) **este {Ɵ(1)}**

i.colectie ← c

i.curent ← c.prim

**sfCreează**

**subalgoritmul** *element*(i, e) **este {Ɵ(1)}**

e ← i.colectie.colectie[i.curent].element

**sfElement**

**funcția** *valid*(i) **este {Ɵ(1)}**

***valid***← i.curent ≠ -1

**sfValid**

**subalgoritmul** *următor*(i) **este {Ɵ(1)}**

i.curent ← i.colectie.colectie[i.curent].urmator

**sfUrmător**

**subalgoritmul** *anterior*(i) **este {Ɵ(1)}**

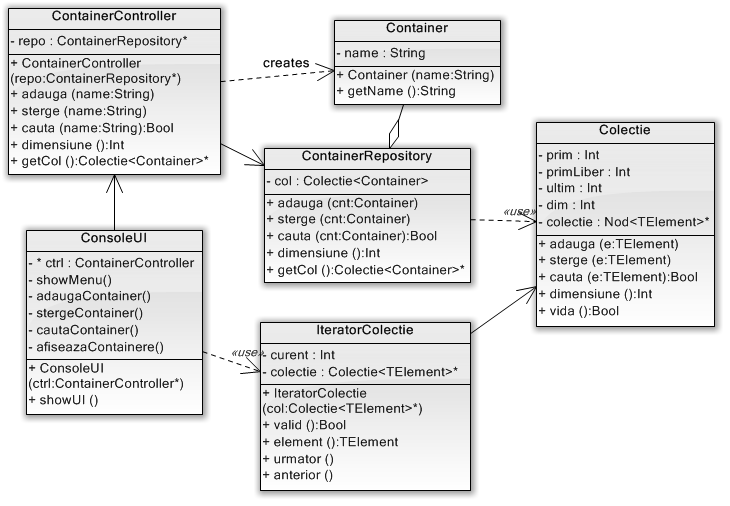
i.curent ← i.colectie.colectie[i.curent].anterior

**sfAnterior**

# Aplicația (enunț și pseudocod)

Creați o aplicație care gestionează containerele dintr-un depozit de fructe. Fructele sunt depozitate in containere, pentru fiecare tip existând un singur model (două containere pentru mere vor fi mereu identice, dar unul pentru mere va fi diferit de unul pentru cireșe). Din acest motiv containerele vor fi identificate prin numele fructelor depozitate în acestea.

Diagrama



În continuare este pseudocodul pentru subalgoritmii care interacționează cu TAD-ul. Se poate observa și în diagramă că toți ceilalți algoritmi îi folosesc pe aceștia ca intermediari.

ContainerRepository

**subalgoritmul** *adaugă*(col , cnt) **este {Ɵ(1)}**

*{cnt:* ***Container;*** *col:* ***Colectie****}*

col.adaugă(cnt)

**sfAdaugă**

**subalgoritmul** *șterge*(col , cnt) **este {O(n)}**

*{cnt:* ***Container;*** *col:* ***Colectie****}*

col.șterge(cnt)

**sfȘterge**

**functia** *caută*(col , cnt) **este {O(n)}**

*{cnt:* ***Container;*** *col:* ***Colectie****}*

***caută*** ← col.caută(cnt)

**sfCaută**

**functia** *dimensiune*(col) **este {Ɵ(1)}**

*{col:* ***Colectie****}*

***dimensiune*** ← col.dimensiune(col)

**sfDimensiune**

ConsoleUI

**subalgoritmul** *afișeazăContainere*() **este {O(n)}**

{ctrl: ↑**ContainerController**; it: **IteratorColecție**}

**dacă** [ctrl].dimensiune() = 0 **atunci**

**tipărește** "Depozitul este gol..."

**altfel**

it ← [[ctrl].getCol()].iterator()

**câtTimp** it.valid()

**tipărește** it.element().getName()

it.urmator();

**sfCâtTimp**

**sfDacă**

**sfAfișeazăContainere**