ADT_DCLL: DoubleCircularLinkedList Specificatie datastructuren college3

Elementen:

Elementen zijn lijst, knopen, en hulpwijzers;

De lijst wijst naar een record met administratieve gegevens rond de lijst met knopen;

elke knoop bevat een data_element

elke knoop bevat pointers naar een linker en een rechter buurknoop;

elke hulpwijzer geeft het adres van een knoop;

```
Type: LijstPointer = ^DoubleCircularLinkedList;
      DoubleCircularLinkedList = record
                                   HeadKnoop:KnoopPointer;
                                   CurrentKnoop: KnoopPointer
                                 end;
      KnoopPointer = ^Knoop;
      Knoop = record
                LeftKnoop:KnoopPointer;
                Data_Element: DataType;
                RightKnoop: KnoopPointer
              end;
```

Structuur: relatie is lineair modulo de omvang van de lijst, RightKnoop wijst naar de opvolger; LeftKnoop wijst naar de voorganger.

Twee extra lijst pointers wijzen naar een begin knoop (HeadKnoop) en huidige knoop (CurrentKnoop), nil als lijst leeg.

Domein: het aantal knopen in de lijst is eindig [0..Max) # open einde afhankelijk beschikbaar geheugen

Operaties: een 12 tal; operaties vinden meestal plaats bij of rond CurrentKnoop; bij elke operatie moeten wijzigingen in de HeadKnoop en CurrentKnoop worden aangegeven

Procedure **FindFirst**(var DCLL:DoubleCircularLinkedList); # bijwerking interne wijzer

Postconditie: CurrentKnoop wijst naar begin knoop (net als HeadKnoop).

Vervolg **ADT_DoubleCircularLinkedList** specificatie datastructuren college 3 bladzijde 2

Procedure **FindRightOne**(var DCLL:DoubleCircularLinkedList); # bijwerking interne wijzer

Preconditie: lijst is niet leeg

Postconditie: de volgende knoop rechtswaar de RightKnoop van de CurrentKnoop naar verwijst wordt de CurrentKnoop.

Procedure **FindLeftOne**(var DCLL:DoubleCircularLinkedList); # bijwerking interne wijzer

Preconditie: lijst is niet leeg

Postconditie: de volgende knoop links waar de LeftKnoop van de CurrentKnoop naar verwijst wordt de CurrentKnoop.

Procedure **RetrieveThisOne**(var DCLL:DoubleCircularLinkedList;var elem:DataType);

Preconditie: linked list is niet leeg.

Postconditie: elem heeft als waarde die van Data_Element in CurrentKnoop gekregen.

Procedure **UpdateThisOne(**var DCLL:DoubleCircularLinkedList;var elem:DataType);

Preconditie: linked list is niet leeg.

Postconditie: Data_Element CurrentKnoop heeft waarde elem gekregen.

Procedure **InsertAfterThisOne**(var DCLL:DoubleCircularLinkedList;var elem:DataType);

Preconditie: er is nog geheugen voor een nieuwe knoop

Postconditie: een nieuwe knoop met als Date_Element elem en als RightKnoop waarde die van de huidige (oude) CurrentKnoop wordt toegevoegd; als LeftKnoop komt een pointer naar de oude CurrentKnoop; de RightKnoop van de oude CurrentKnoop wijst naar de nieuwe knoop; CurrentKnoop wijst naar deze nieuwe knoop; als HeadKnoop nil was krijgt deze ook de waarde van CurrentKnoop.

Procedure **InsertBeforeThisOne**(var DCLL:DoubleCircularLinkedList;var elem:DataType);

Preconditie: er is nog geheugen voor een nieuwe knoop

Postconditie: een nieuwe knoop met als Date_Element elem en als LeftKnoop waarde die van de huidige (oude) CurrentKnoop wordt toegevoegd; als RightKnoop komt een pointer naar de oude CurrentKnoop; de LeftKnoop van de oude CurrentKnoop wijst naar de nieuwe knoop; CurrentKnoop wijst naar deze nieuwe knoop; als HeadKnoop nil was krijgt deze ook de waarde van CurrentKnoop.

Vervolg **ADT_DoubleCircularLinkedList** specificatie datastructuren college 3 bladzijde 3

Procedure **DeleteThisOne(var DCLL:DoubleCircularLinkedList)**;

Preconditie: linked list is niet leeg

Postconditie: CurrentKnoop verwijderd uit de lijst; Voorganger's RightKnoop moet te deleten RightKnoop pointerwaarde overnemen, Opvolger's LeftKnoop moet LeftKnoop wijzer van te deleten knoop overnemen; CurrentKnoop wijst naar deze opvolger (of voorganger); als CurrentKnoop gelijk aan HeadKnoop was, wordt CurrentKnoop's NextKnoop zowel HeadKnoop als CurrentKnoop (eventueel dus beide nil als er maar 1 knoop in de lijst zat).

Function **Empty(var DCLL:DoubleCircularLinkedList):**boolean;

Postconditie: Y als linked list leeg (head_pointer=nil) anders N

Function **Last(var DCLL:DoubleCircularLinkedList)**:boolean; #is CurrentKnoop laatste in lijst?

Postconditie: Y als next_pointer van CurrentKnoop HeadKnoop is, anders N

Procedure **Create(var DCLL:DoubleCircularLinkedList)**;

Postconditie: een lege linked list wordt aangemaakt; HeadKnoop en CurrentKnoop zijn nil.

Procedure **Delete(var DCLL:DoubleCircularLinkedList)**;

Postconditie: geheugen voor DCLL weer vrijgegeven.