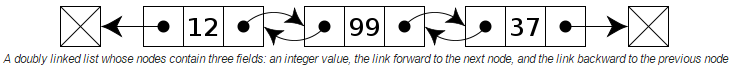
Doublylistedlist

En doublylistedlist er blevet valgt til projektet da det er blevet vurderet at den har de fleste fordele i forhold til andre datastrukturer. Som der er erfaring med i gruppen, en fordel ved at holde sig til en datastruktur er mere tid til at udvikle den og lave den rigtig god i forhold til at lave mange forskellige strukturere. En doubly linked

Et virsualt exmple på en doubly listed list



Fra wikipedia 01-04-2015. Denne kommer til at ha samme struktur som billede lige over men i stedet for at kun gemme ints lavers den som en template klasse så vi kan gemme alle data typer.

**void headInsert(T info);**

Parameter **T info**: Datatypen som skal gemmes.

Return: ingen

Beskrivelse: Bruges til at data indsætte i head af listen.

**void tailInsert(T info);**

Parameter **T info**: Datatypen som skal gemmes.

Return: ingen

Beskrivelse: Bruges til at indsætte i tail af listen.

**void headRemove();**

Parameter: ingen

Return: ingen

Beskrivelse: Bruges til at slette head dataen.

**void tailRemove();**

Parameter: ingen

Return: ingen

Beskrivelse: Bruges til at slette tail dataen.

**int Search(T info);**

Parameter **T info**: Det data den skal finde i listen.

Return **INT**: placering i listen.

Beskrivelse: Bruges finde data i listen.

**Void Delete(int place);**

Parameter **T info**: Det data den skal finde i listen.

Return **INT**: placering i listen.

Beskrivelse: Bruges til at slet data i listen.

**void cop1y(DoublyListedList\* sourceDoublyListedList);**

*Private*

Node\* head

Beskrivelse: Head ponteren som bruges internt i klassen, og peger på det første element i listen.

Node\* tail

Beskrivelse: Tail ponteren som bruges internt i klassen, og peger på det bagerste element i listen.

Node

Beskrivelse: En klasse som har 3 ting, next pointer, previous pointer og info som er en template data type.