

Imperativ Programmering og Datastrukturer

Implementaion af hængtede lister (køer og stakke)

René Rydhof Hansen

6. december 2009

Mål

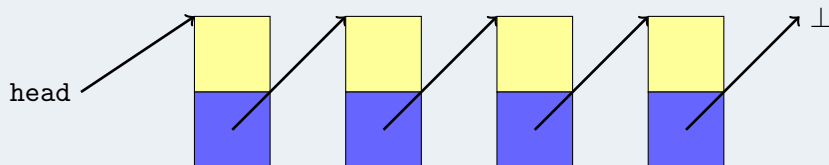
- At kunne implementere og anvende linked lists
- At kunne implementere og anvende stakke og køer

Hægtede lister

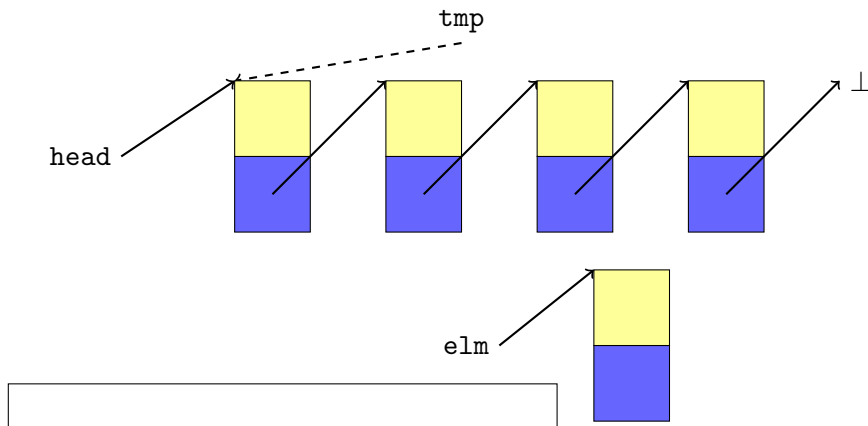
Elementer

```
class Record
{
    String titel;
    Record next;
}
```

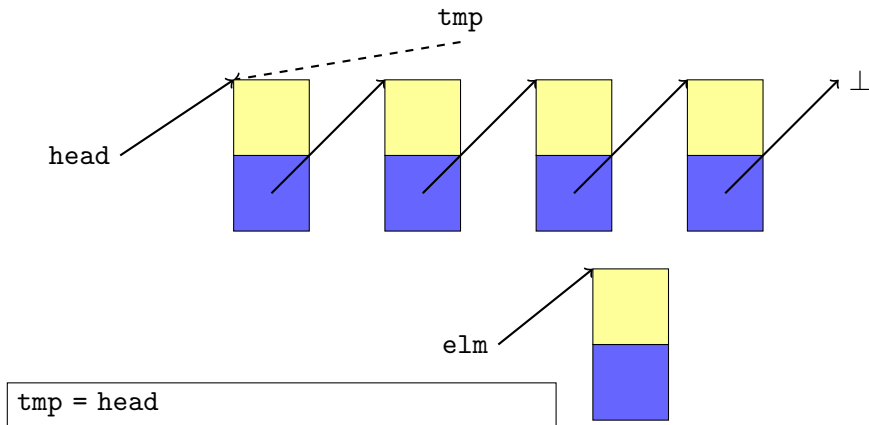
Liste af elementer



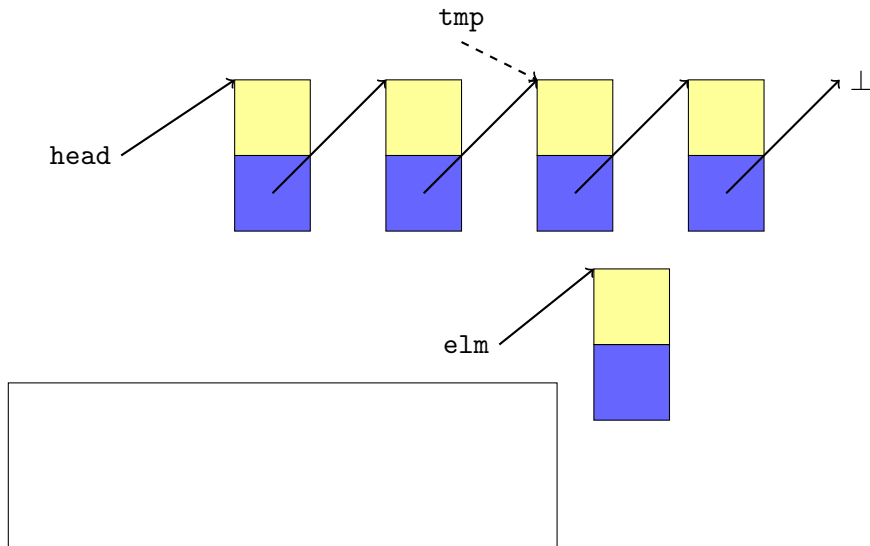
Indsættelse af element i hægtet liste



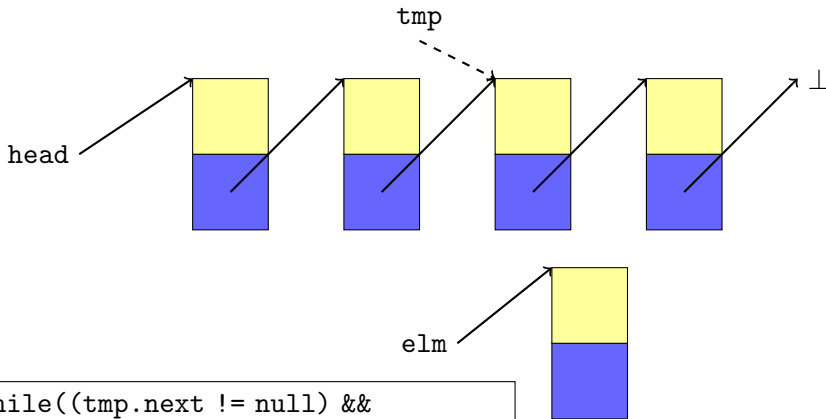
Indsættelse af element i hægtet liste



Indsættelse af element i hægtet liste

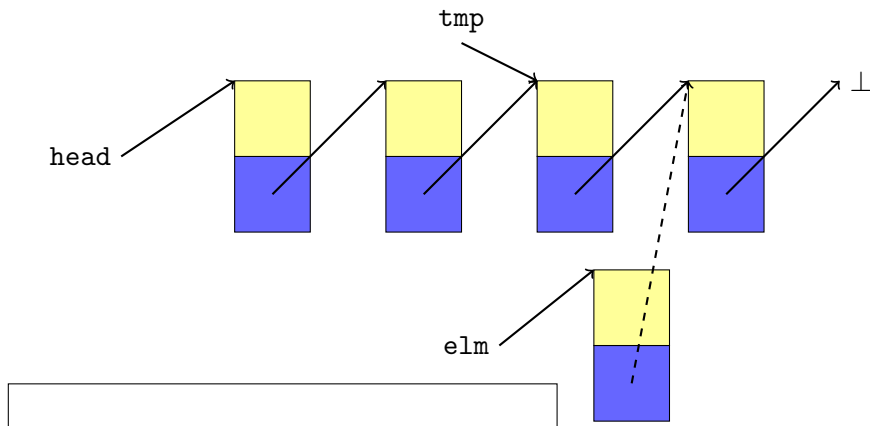


Indsættelse af element i hægtet liste

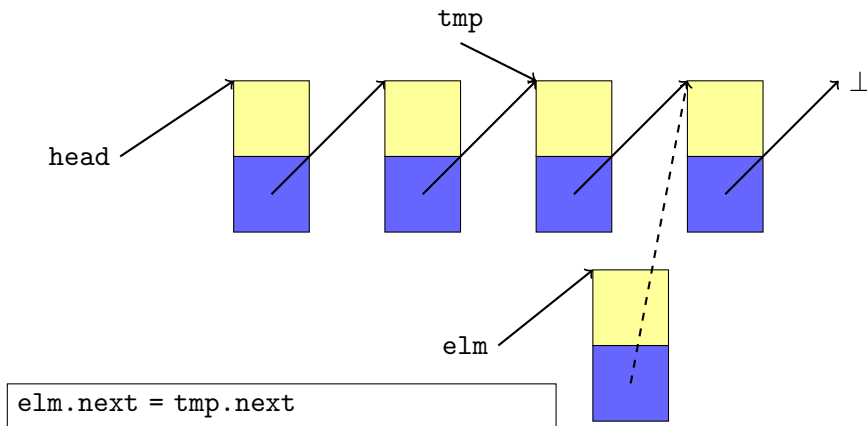


```
while((tmp.next != null) &&  
      (tmp.next.title < elm.title) {  
    tmp = tmp.next;  
}
```

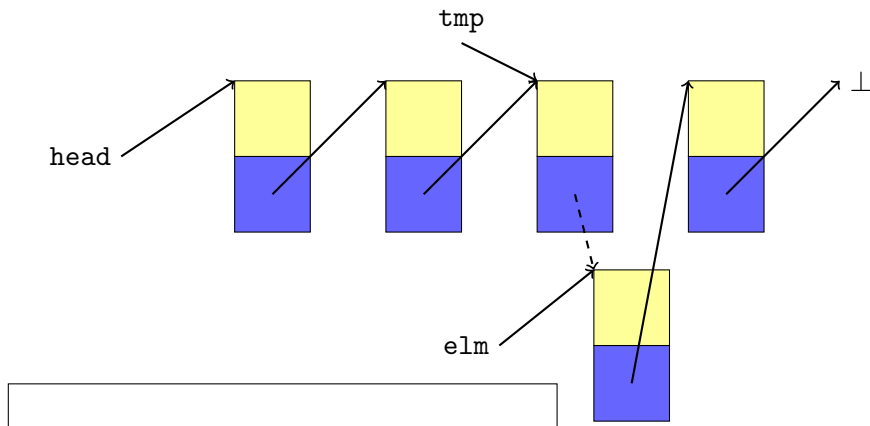
Indsættelse af element i hægtet liste



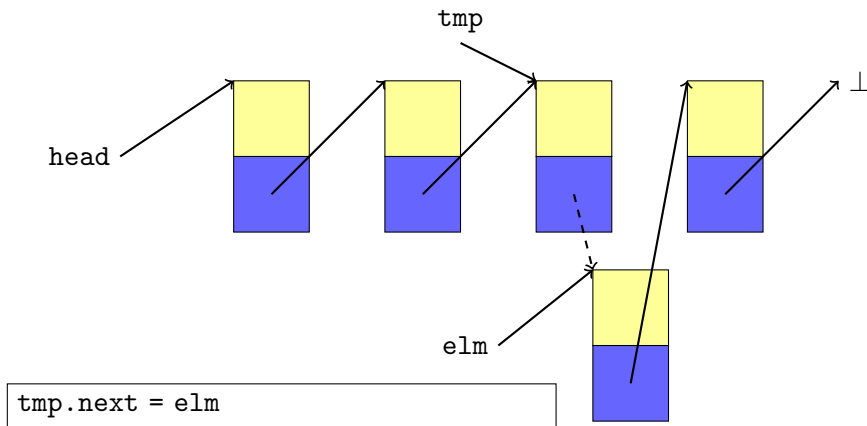
Indsættelse af element i hægtet liste



Indsættelse af element i hægtet liste



Indsættelse af element i hægtet liste



Indsættelse af element i hægtet liste (iterativt)

```
static void orderedInsert(Album elm) {
    Album tmp;

    tmp = head;
    while((tmp.next != null) &&
           (tmp.next.title < elm.title)) {
        tmp = tmp.next;
    }
    elm.next = tmp.next;
    tmp.next = elm;
}
```

Indsættelse af element i hægtet liste (iterativt)

```
static void orderedInsert(Album elm) {
    Album tmp;

    if((head == null) ||
        (elm.title < head.title)) {
        elm.next = head;
        head = elm;
    } else {
        tmp = head;
        while((tmp.next != null) &&
            (tmp.next.title < elm.title)) {
            tmp = tmp.next;
        }
        elm.next = tmp.next;
        tmp.next = elm;
    }
}
```

Indsættelse af element i hægtet liste (iterativt)

```
static void orderedInsert(Album elm) {
    Album tmp;

    if((head == null) ||
        (elm.title < head.title)) {
        elm.next = head;
        head = elm;
    } else {
        tmp = head;
        while((tmp.next != null) &&
            (tmp.next.title < elm.title)) {
            tmp = tmp.next;
        }
        elm.next = tmp.next;
        tmp.next = elm;
    }
}
```

Indsættelse af element i hægtet liste (iterativt)

```
static void orderedInsert(Album elm) {
    Album tmp;

    if((head == null) ||
        (String.Compare(elm.title,head.title) < 0)) {
        elm.next = head;
        head = elm;
    } else {
        tmp = head;
        while((tmp.next != null) &&
            (String.Compare(tmp.next.title,elm.title) < 0)) {
            tmp = tmp.next;
        }
        elm.next = tmp.next;
        tmp.next = elm;
    }
}
```

Indsættelse af element i hægtet liste (rekursivt)

```
static Album recInsert(Album head, Album elm) {  
    if(head == null) {  
        head = elm;  
        elm.next = null;  
    } else if (String.Compare(elm.title, head.title) <  
        elm.next = head;  
        head = elm;  
    } else {  
        head.next = recInsert(head.next, elm);  
    }  
    return head;  
}
```


Indsættelse af element i hægtet liste (rekursivt)

```
static void recInsert(ref Album head, Album elm) {
    if(head == null) {
        head = elm;
        elm.next = null;
    } else if (String.Compare(elm.title, head.title) <
        elm.next = head;
        head = elm;
    } else {
        recInsert(ref head.next, elm);
    }
}
```

Stak

- Last In, First Out (LIFO) / First In, Last Out (FILO)
- Operationer
 - Push (indsæt top-element)
 - Pop (fjern top-element)

Kø

- Last In, Last Out (LILO) / First In, First Out (FIFO)
- Operationer
 - Put (indsæt nyt bund-element)
 - Get (fjern top-element)

- `reversi.cs`: reversering af liste
- `ll-insert.cs`: indsættelse i liste
- `ll-insert2.cs`: indsættelse i lister (generelt)
- `ll-insert-rec.cs`: indsættelse i liste (rekursivt)

Opsummering og Næste gang

Opsummering

- Implementation af hægtet liste (ordnet indsættelse)

Næste gang

- Gennemgang af afleveringsopgave 3
- Implementation af hægtet liste