Föreläsning 4

# Agenda

ADT (abstrakta datatyper)

Listor – arraylista, nodlista, sekvenser i c++

ADT stack – tillämpningar, stackar i c++

ADT kö – kö i c++

**ADT**

*Abstrakt datatyp*

Maskinoberoende högnivåbeskrivning av data och operationer på data, t.ex. stack, kö, mängd

*Datastruktur*

Logisk organisation av datorns minne för att lagra data

*Algoritm*

Högnivåbeskrivning av konkreta operationer på datastrukturer

ADT implementeras med lämpliga datastrukturer och algoritmer

Program implementerar algoritmer och datastrukturer i något visst språk

**Lista**

Arraylista använder index

Nodlista använder position (t.ex. första element i Listan, nästa position, föregående etc.)

**Arraylista**

Operationer:

Size()

Isempty()

elemAtIndex(i)

setAtIndex(i, x) returnera element med index i och ersätt det med x som nytt element med index i

insertAtIndex(i, x) sätt in, utöka storleken

**Nodlista**

även size och isempty..

first(), last()

prev(p)

next(p) returnerar positionen för elementet som följer på p

set(p,x)

insertFirst(x)

**sekvenser i c++**

Vector motsvarar arraylist på ett bra sätt

Forward\_list ger oss möjlighet att få tag på nästa element

List ger oss möjlighet att få båda hållen, likt nodlista

**Stack**

top(s) returnerar det översta elementet i stack S

pop(s) tar bort översta elementet

push(s,x) lägger x överst

makeemptystack() skapar en ny tom stack

isemptystack(s) kollar om den är tom

inga iteratorer

Finns ingen .clear() på stack så då får man skriva en egen sån funktion.

Samma gäller med .pop() som inte returnerar som det borde, utan returnerar void.

**Kö**

Först in först ut

I c++:

q.size()

q.empty()

q.front()

q.push(x)

q.pop()