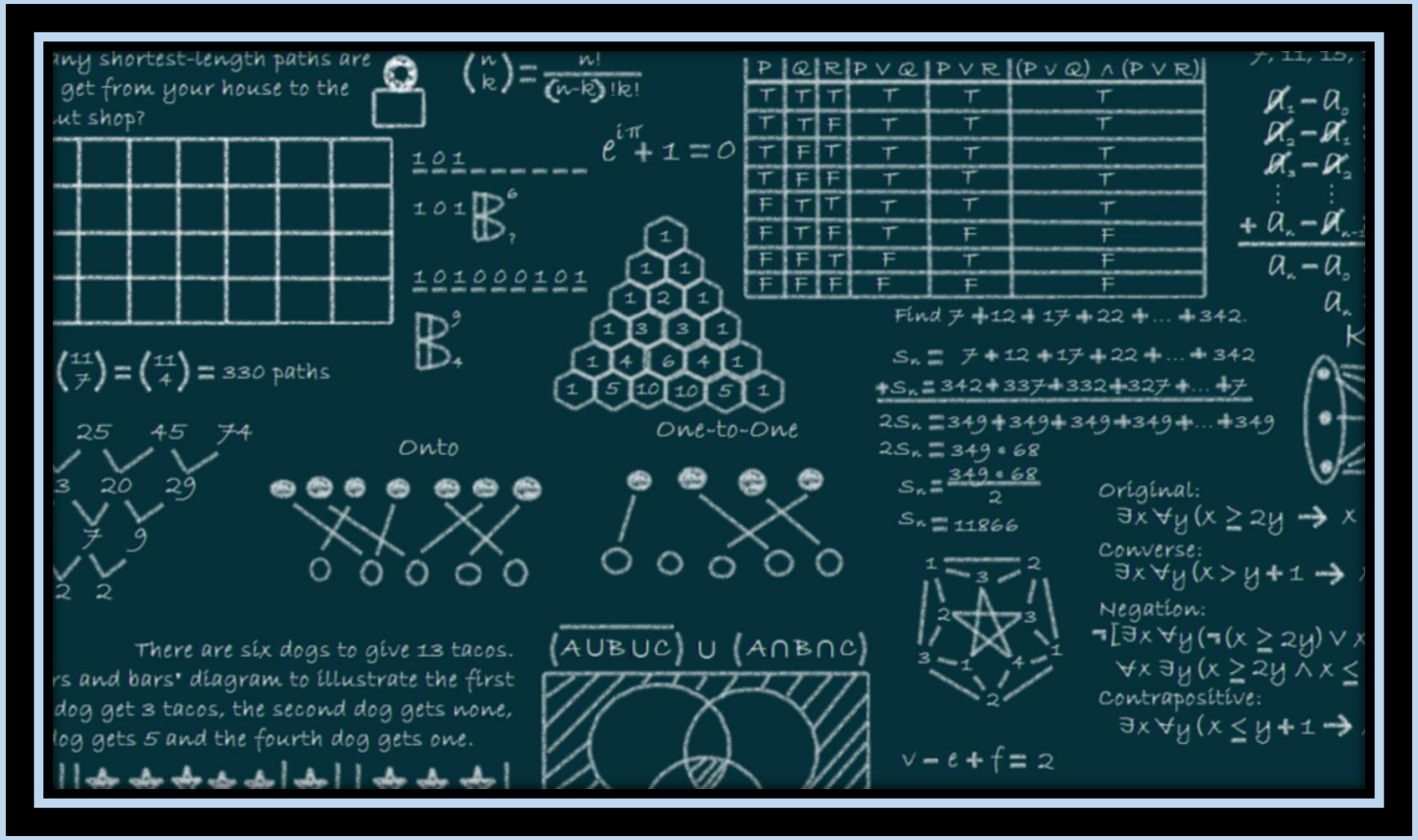


KURSINTRO

Datastrukturer och algoritmer
KYH – 2022 HT

Andreas Nilsson Ström

Kursintro

- Syfte och mål
- Schema & Examination
- Betygskriterier

Syfte & Mål

Kunskaper om/i

- Begreppen abstrakt datatyp, rekursion och komplexitet
- Hur algoritmer definieras och byggs upp för att lösa definierade problem
- Användningen av tekniker för algoritmdesign på diskreta problem

Färdigheter om/i

- Bedöma datastrukturers och algoritmers lämplighet för olika typer av problem inom IoT-området

Kompetenser om/i

- Kunna anpassa praktiska algoritmer och datastrukturer för att lösa och implementera givna problem korrekt och på ett effektivt sätt
- Förstå hur man utvecklar, implementerar och anpassar datastrukturer och algoritmer för en given situation inom IoT

Syfte & Mål

Med andra ord

- Kunskaper om begrepp och tekniker
- Läran oss om olika datastrukturer och algoritmer
- Skriva program som effektiv löser problem

Schema

- 4 veckor
  - Totalt 10 dagar (inkl. tentadagen)
- Examination?
  - 3 inlämningar – En vecka per uppgift
  - En skriven tenta: Torsdag, 1 december

Betygskriterier - Från kursplanen

Betyg sätts i form av **Icke godkänt** (IG), **Godkänt** (G) eller **Väl godkänt** (VG)

Icke godkänt (IG)

- Den studerande har fullföljt kursen men inte nått alla mål för kursen.

Betygskriterier – Godkänt (G)

För betyget Godkänt ska den studerande…

- Redogöra för begreppen abstrakt datatyp, rekursion och komplexitet.
- […] ha kunskap om hur algoritmer definieras och byggs upp för att lösa definierade problem samt ha kännedom om användningen av tekniker för algoritmdesign på diskreta problem.
- […] kunna bedöma datastrukturers och algoritmers lämplighet för olika typer av problem inom IoT-området.
- […] kunna anpassa praktiska algoritmer och datastrukturer för att lösa och implementera givna problem korrekt och på ett effektivt sätt
- samt förstå hur man utvecklar, implementerar och anpassar datastrukturer och algoritmer för en given situation inom IoT.

Betygskriterier – Godkänt (G)

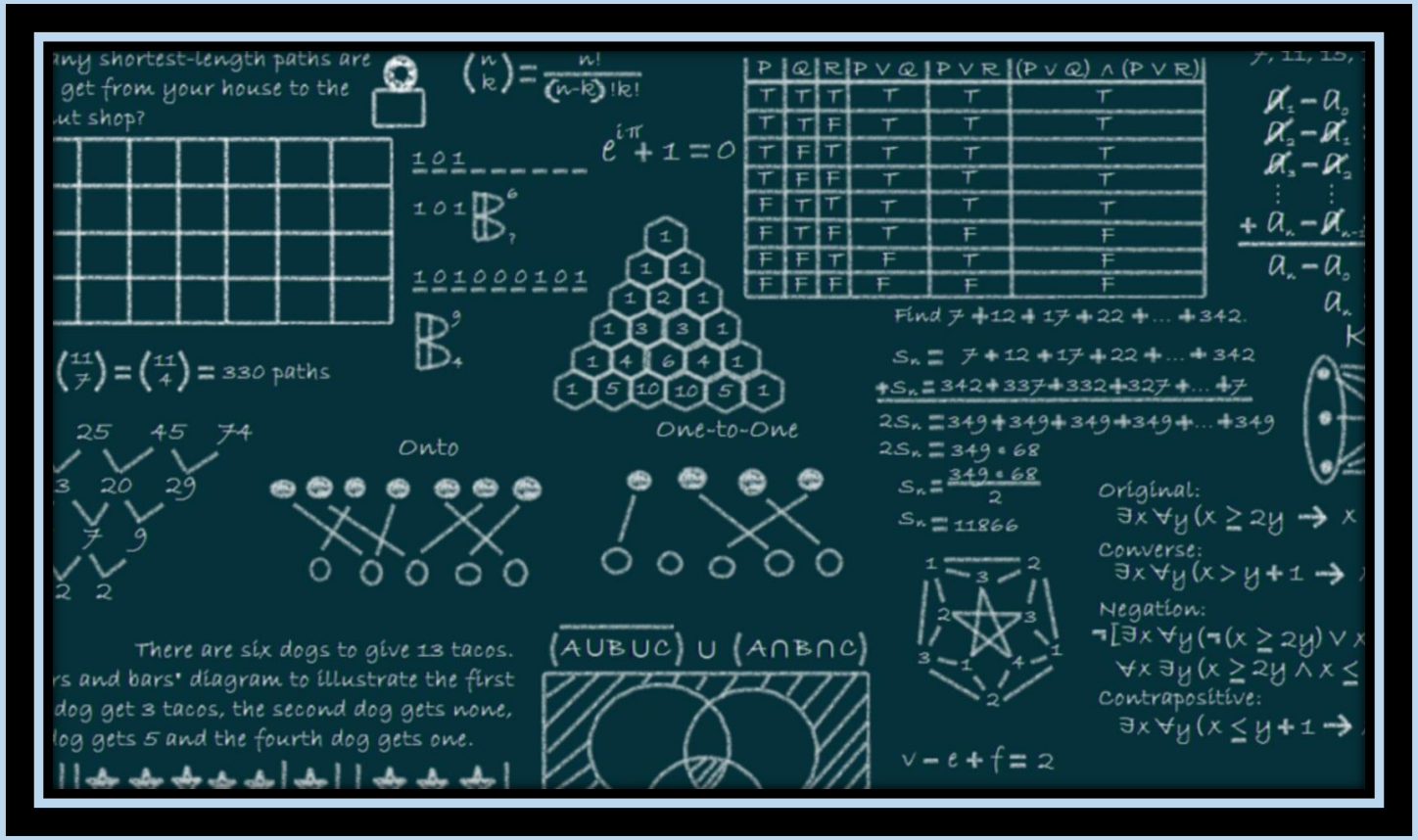
För betyget Godkänt ska den studerande…

- Redogöra för begreppen abstrakt datatyp, rekursion och komplexitet.
- […] ha kunskap om hur algoritmer definieras och byggs upp för att lösa definierade problem samt ha kännedom om användningen av tekniker för algoritmdesign på diskreta problem.
- […] kunna bedöma datastrukturers och algoritmers lämplighet för olika typer av problem inom IoT-området.
- […] kunna anpassa praktiska algoritmer och datastrukturer för att lösa och implementera givna problem korrekt och på ett effektivt sätt
- samt förstå hur man utvecklar, implementerar och anpassar datastrukturer och algoritmer för en given situation inom IoT.

Betygskriterier – Väl Godkänt (VG)

Förutom kriterierna för betyget Godkänt så ska den studerande

- självständigt med säkerhet kunna bedöma datastrukturers och algoritmers lämplighet för olika typer av problem inom IoT-området
- även kunna självständigt anpassa praktiska algoritmer och datastrukturer för att lösa och implementera givna problem korrekt och effektivt



FÖRELÄSNING 1

Datastrukturer och algoritmer
KYH – 2022 HT

Andreas Nilsson Ström

Agenda

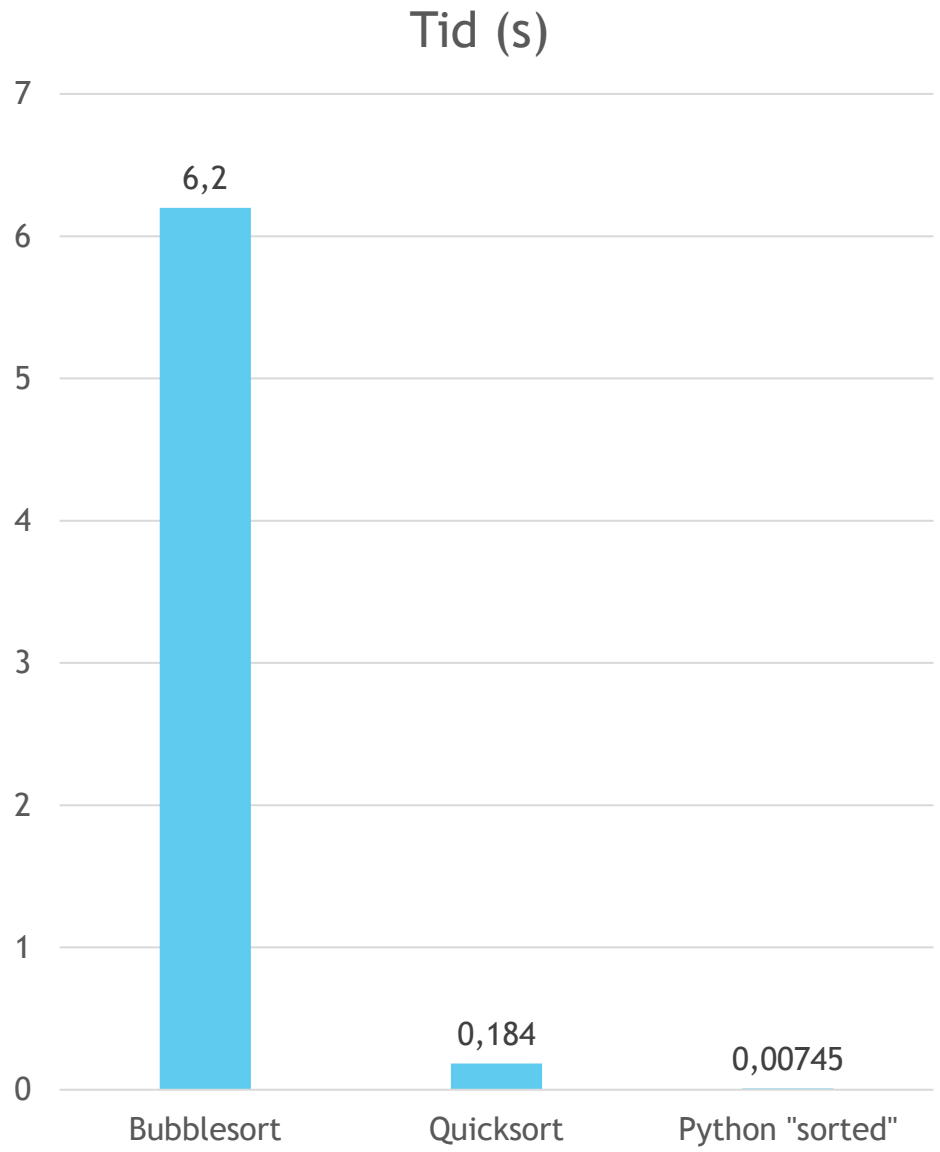
- Vad är datastrukturer och algoritmer?
- Översikt över kursen och ämnet
- Repetition av nyckelkoncept från programmeringskursen
- (Repetition) Intro till vad som händer bakom kulisserna i en dator
- Första datastrukturen
- Övningar
- Intro till första labben

Datastrukturer och algoritmer

- Vad är dessa saker och varför har vi en kurs i dem?
- Datastrukturer = Sätt att lagra data på datorn
- Målet: Att vi ska kunna lösa problem på ett effektivt sätt
  - Om vi inte ser oss för så slösar vi tid och resurser

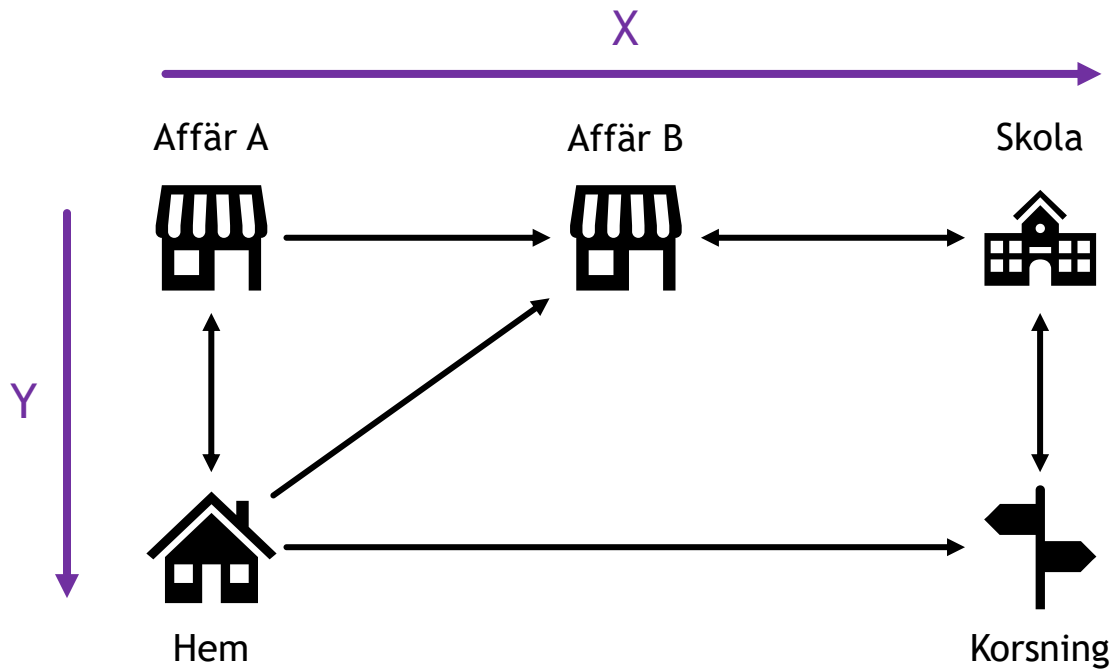
Demo: Sortering av tal i Python

Summering av demo



- Skapar en lista med 1000 heltal
  - (mellan 0 och 10 000)
- Sorterar listan på tre olika sätt
- Medelvärdet av 100 sorteringar

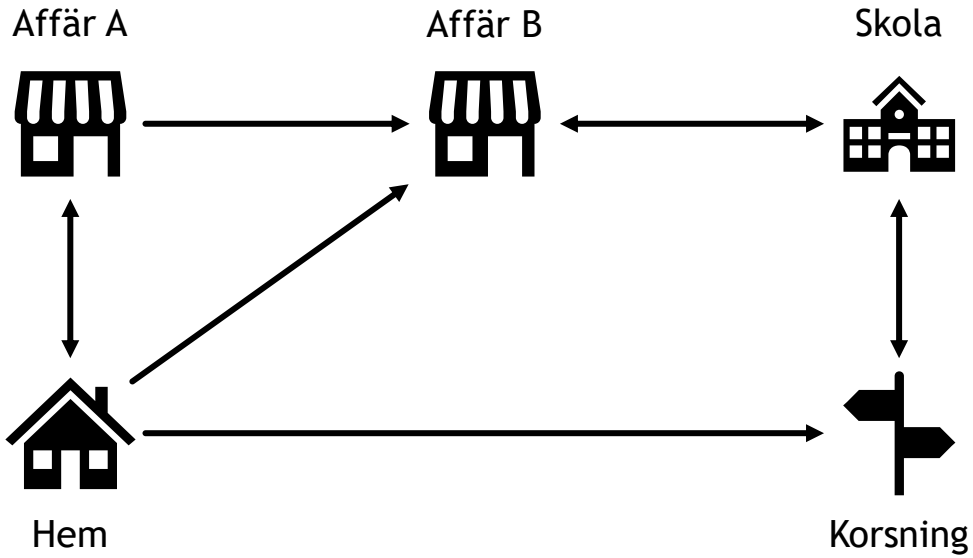
Exempel på ett praktiskt problem



Vägar
(Hem, Affär A)
(Hem, Affär B)
(Hem, Korsning)
(Affär A, Affär B)
(Affär A, Hem)
(Affär B, Skola)
(Skola, Affär B)
(Skola, Korsning)
(Korsning, Skola)

Plats	Koordinater (x, y)
Hem	0, 1
Affär A	0, 0
Affär B	1, 0
Skola	2, 0
Korsning	2, 1

Plats	Vägar
Hem	(Affär A, Affär B, Korsning)
Affär A	(Hem, Affär B)
Affär B	(Skola)
Skola	(Affär B, Korsning)
Korsning	(Skola)



Algoritm:

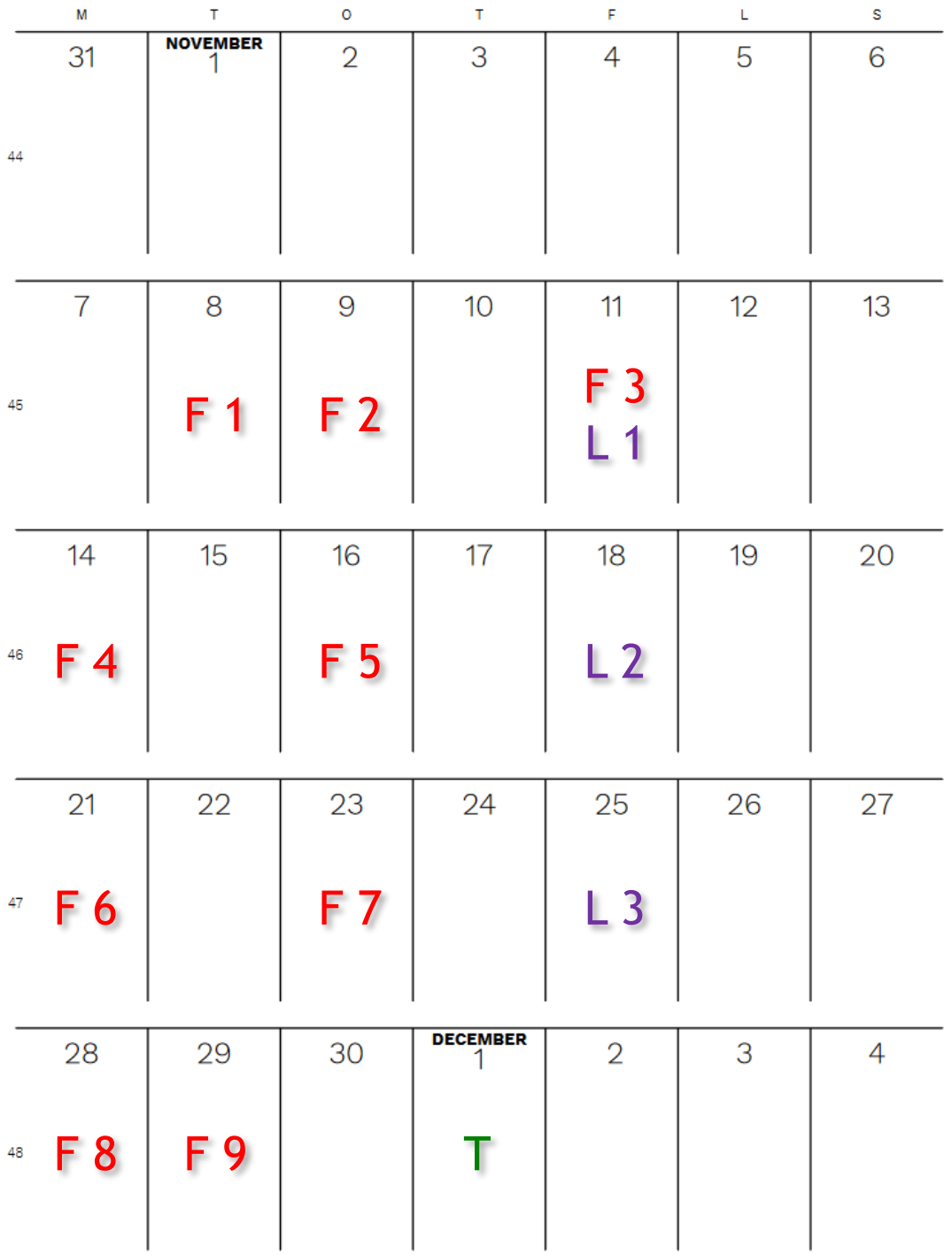
- Börja hemma
- Hitta alla platser du kan ta dig till från hemmet
- För alla platserna, hitta alla vägar dit
- Repetera till du kommer till skolan
- Jämför avståndet du rest
- Hitta kortaste avståndet

Algoritmen påverkar hur vi vill lagra datan!

Översikt

Översikt

- F = Föreläsning
- L = Deadline för labb
- T = Tenta

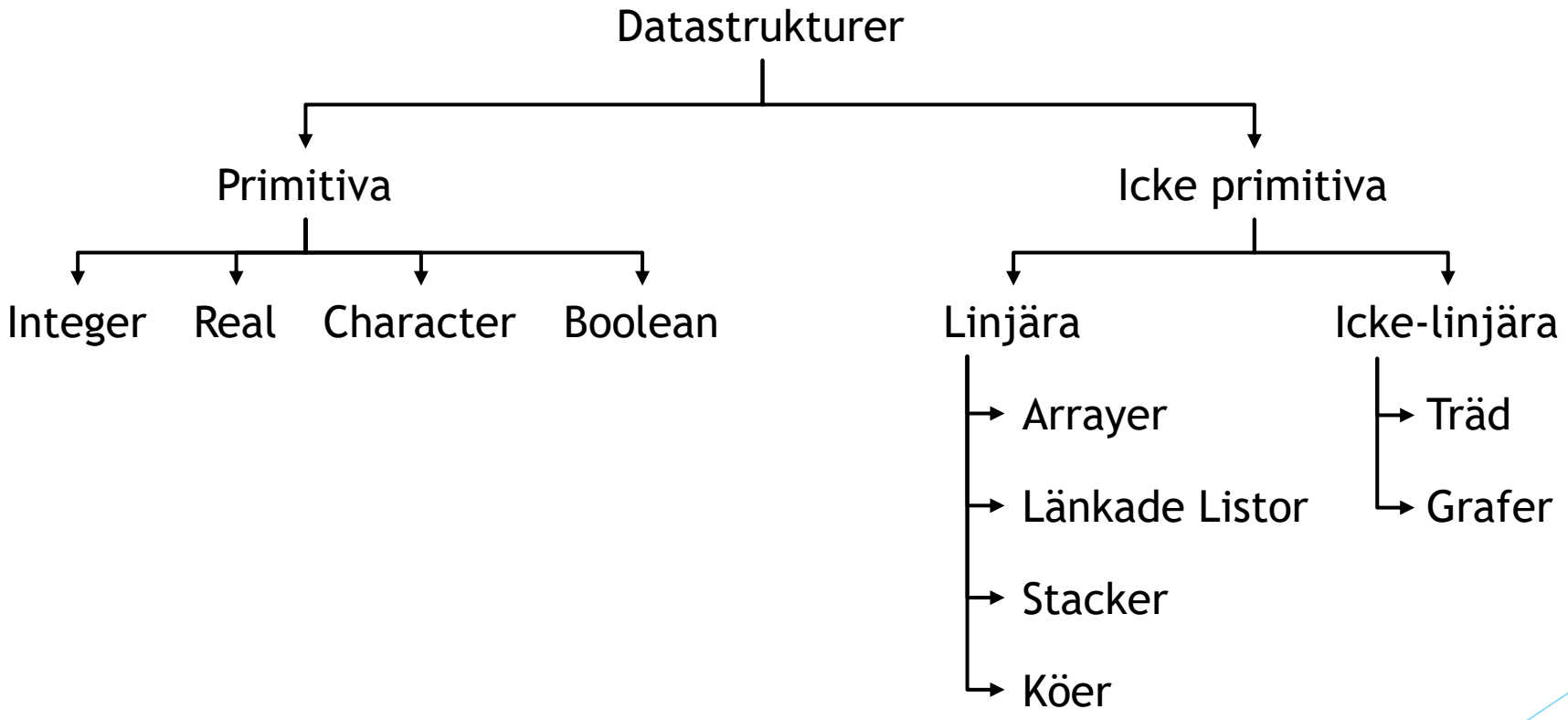


Översikt av ämnen

- Strukturer för lagring av data
- Algoritmer för att söka i, och hantera data

- … Arrayer, träd, köer, tabeller, …

Exempel på datastrukturer



Repetition från förra kursen

Repetition av Python

- Klasser
  - Metoder, __init__, argument, attribut
- Listor
- Dictionaries

Datastruktur: Array

Array

- En samling värden som vi lagrar efter varandra
- Har en fast längd
- Kan (oftast) bara lagra saker av samma typ

- I grunden likt en lista från Python

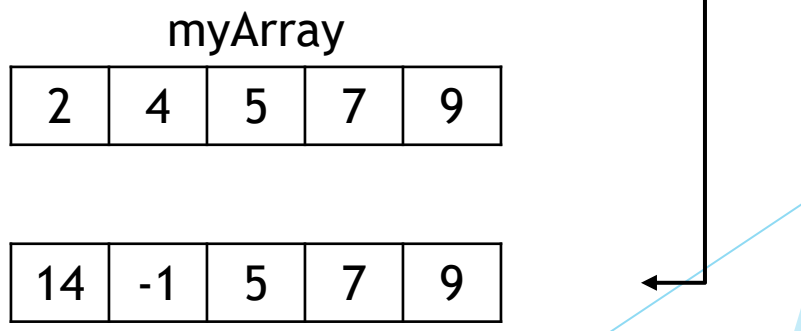
```
[5, 7, -4, 20, 100]
["Hello", "world", "!"]
```

- Exempel i C:

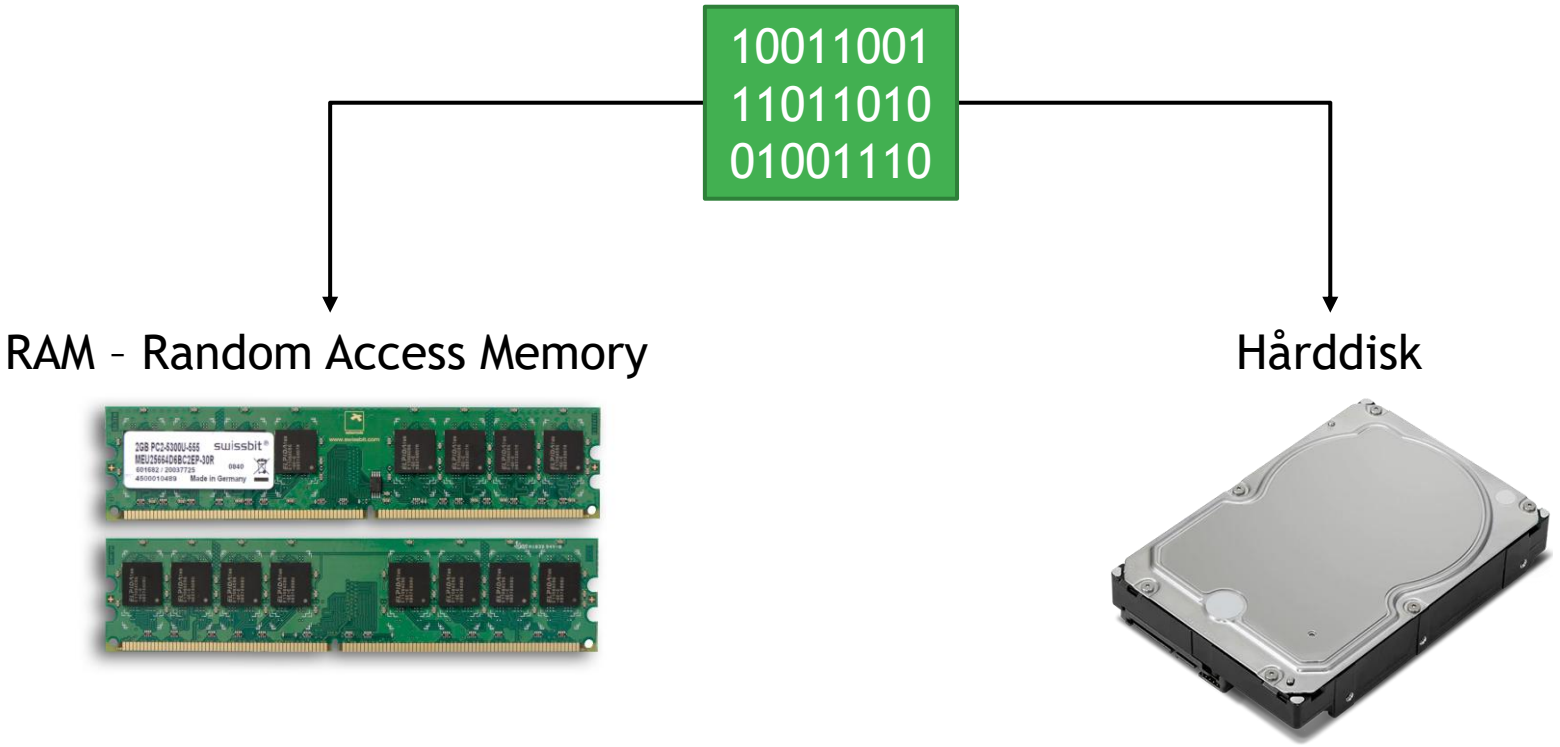
```
int myArray[5] = {2, 4, 5, 7, 9};
```

```
myArray[0] = 14;
myArray[1] = -1;
```

Vad händer om vi vill lägga till två värden?

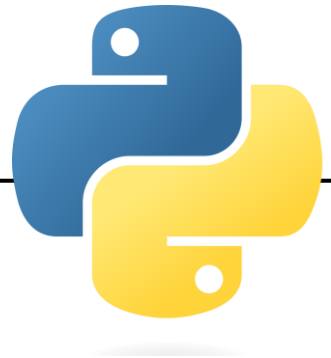


Lagring



	RAM	Hårddisk
Ca. Storlek	8 GB – 32 GB	128 GB – 8 TB
Ca. Hastighet	2-100 GB/s	80-200 MB/s

Lagring

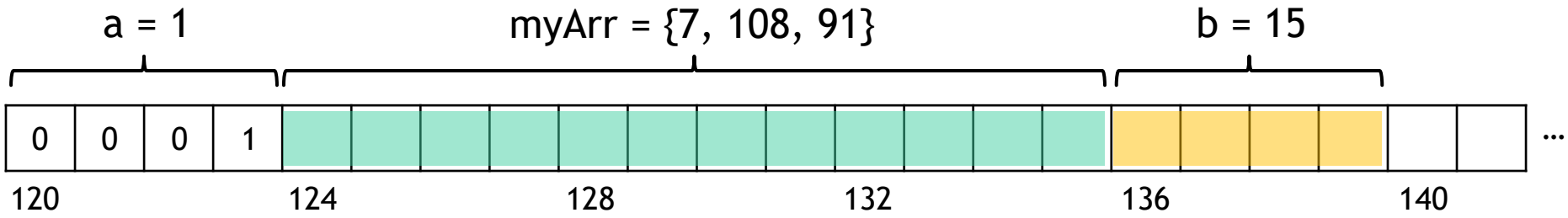


RAM – Random Access Memory

Hårddisk

Lagring i RAM-minnet

- När vi skriver "a = 1" i Python så lagras det i RAM-minnet
- Vårt program blir tilldelat en bit RAM-minne av operativsystemet (Windows, MacOS)



Platser:
10 -> 20 -> 40 -> 80 -> …

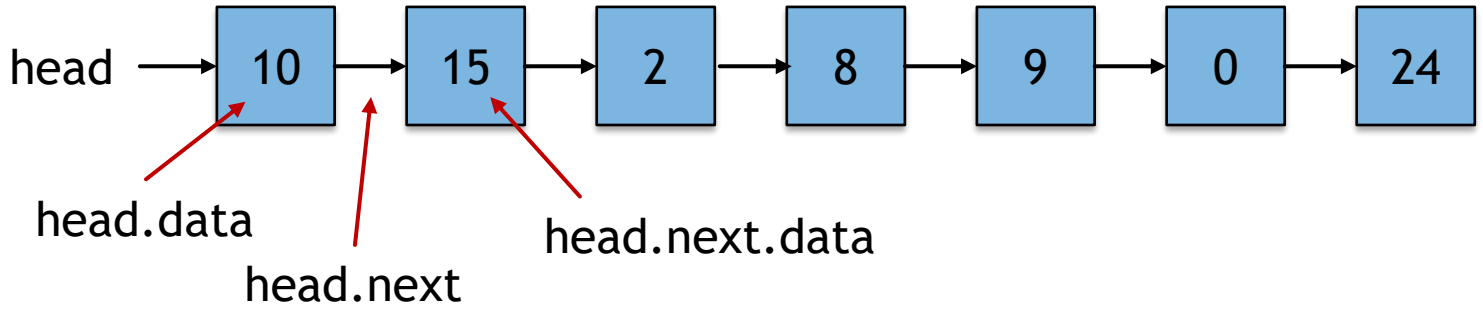
Länkade listor

Länkade listor

- Säg att vi vill lagra en serie siffror
  - 10, 15, 2, 8, 9, 0, 24
- Vi kan så klart använda en array/lista
- En annan variant är att göra en länkad lista. Detta är särskilt användbart när vi inte på förhand vet hur mycket data som ska lagras

Länkade listor

- Om vi börjar med värdena 10, 15, 2, 8
- Vi gör en låda för varje värde
- Varje låda har en pil till nästa värde
- Sen när vi lägger till värden så behöver vi bara göra flera pilar: 9, 0, 24




```
class Box:
    data: int
    next: Box
```

Demo: Länkad lista i Python

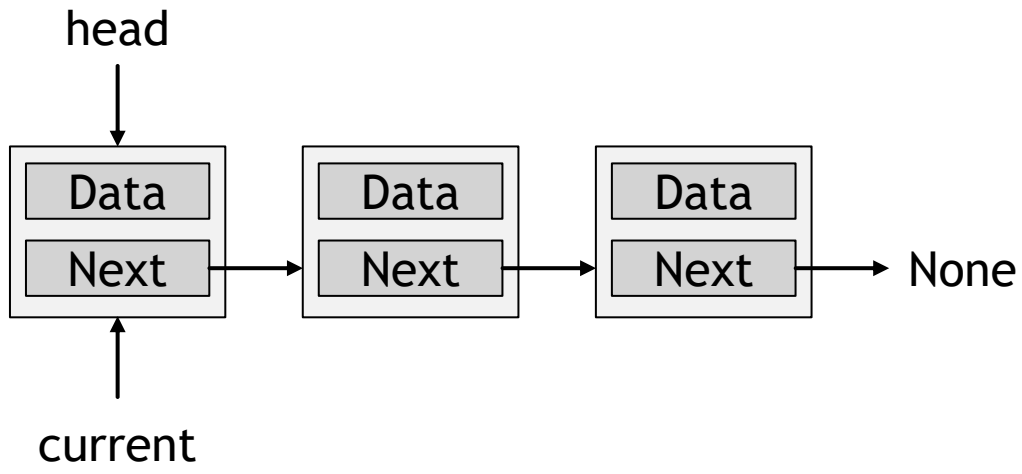
Länkade listor

- Så vi har döpt om Box -> Node

Övning

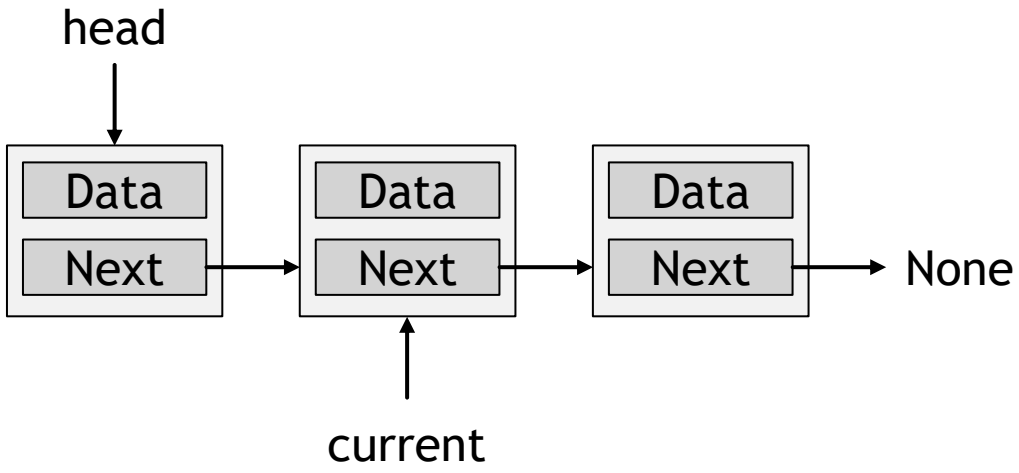
- Skriv en metod i LinkedList-klassen
  - Count(): Som börjar med head-noden och räknar hur många noder det är i listan
  - Sum(): Som summerar alla tal från alla noder i listan
    - (Detta förutsätter att ni lagrar tal!)
  - Add_to_end(value): Lägger till ett värde i slutet av vår lista

"add_to_end()" – även kallad "append()"



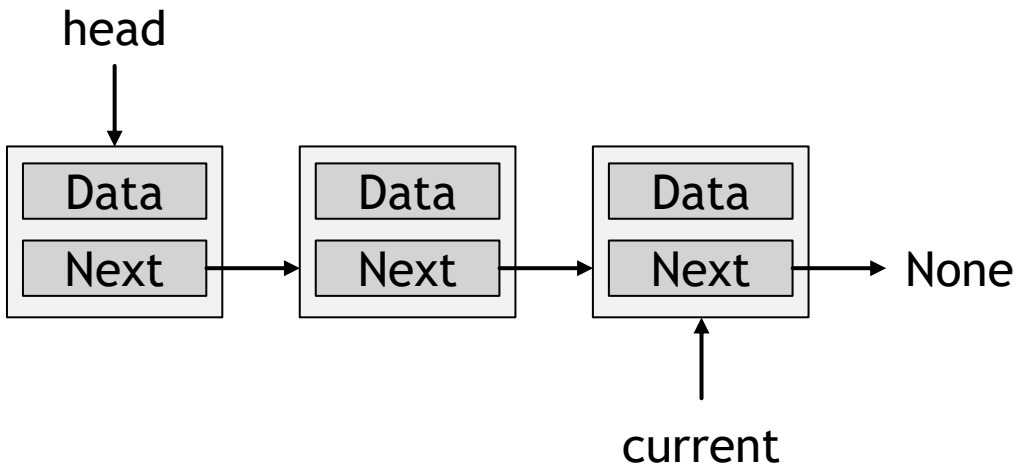

```
def append(data):
    current = self.head
```

"add_to_end()" – även kallad "append()"



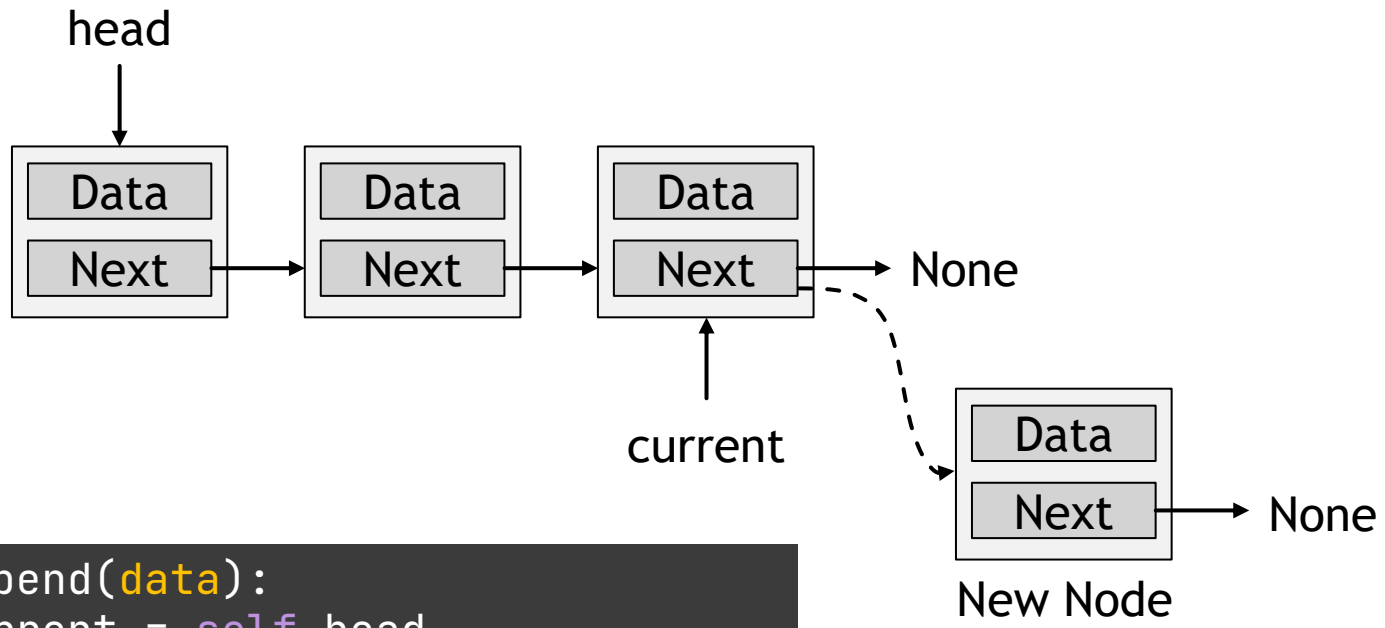

```
def append(data):
    current = self.head

    while current.next is not None:
        current = current.next
```

"add_to_end()" – även kallad "append()"



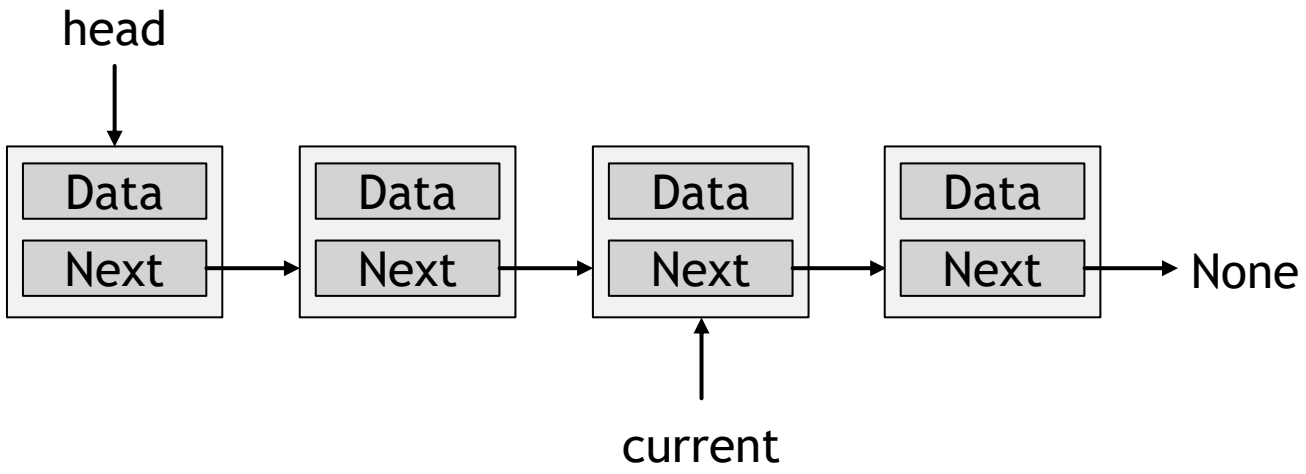

```
def append(data):
    current = self.head

    while current.next is not None:
        current = current.next
```

"add_to_end()" – även kallad "append()"




```
def append(data):
    current = self.head

    while current.next is not None:
        current = current.next

    new_node = Node(data)

    current.next = new_node
```

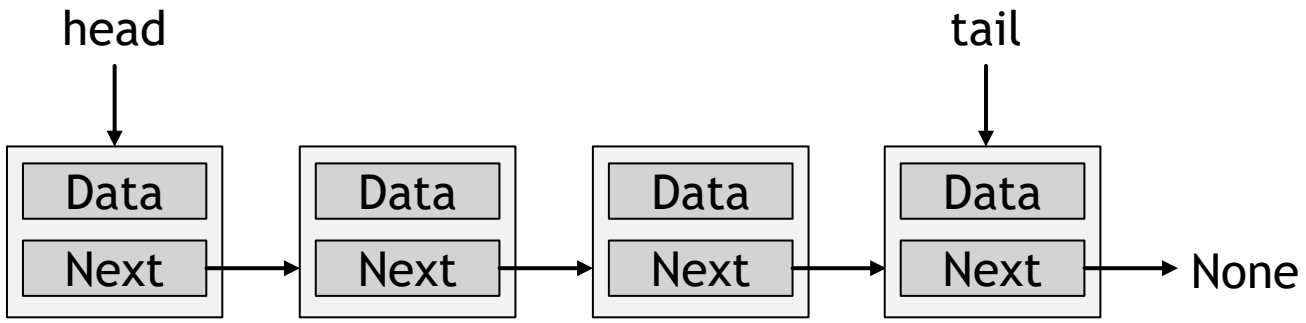
"add_to_end()" - även kallad "append()"




```
def append(data):
    current = self.head

    while current.next is not None:
        current = current.next

    new_node = Node(data)

    current.next = new_node
```

För att göra livet lite enklare…

- Om vi även har en "tail" som alltid pekar på sista Noden…



- **Se min exempelkod för full lösning**

```
def append(data):
    current = self.head

    while current.next is not None:
        current = current.next

    new_node = Node(data)

    current.next = new_node
```

→

```
def append(self, data):
    new_node = Node(data)

    self.tail.next = new_node
```

Veckans inlämning

Veckans inlämning

- Deadline: Fredag kl 23:59
- Labbpartners väljs av er eller genom lottning
- Inlämningen finns att läsa i Omniway

Inlämningsuppgift

- Skapa en Node (med data och next)
  - Ska ha en metod "__str__(self)" som returnerar en strängrepresentation av datan
- Skapa en klass "Storage" som har följande metoder:
  - __init__(self): Sätter "self.head = None"
  - push(self, data): Lägger till en nod i början av listan
  - pop(self): Tar bort första Noden i listan (och returnerar nodens data)
  - peek(self): Returnerar datan från första noden i listan, utan att ta bort noden.
  - isempty(self): Returnerar True om listan är tom, annars False
- Se lärarens GitHub: Assignments/Assignment 1
  - Skelett för uppgiften, och tester som ska fungera innan ni lämnar in

Förklara ord

- Datastruktur
- Algoritm
- Array

Till nästa gång…

- Övningar: Hackerrank
  - https://www.hackerrank.com/domains/data-structures?filters%5Bsubdomains%5D%5B%5D=linked-lists
  - Till och med "Compare two linked lists"
- Mer läsning:
  - https://www.happycoders.eu/algorithms/array-vs-linked-list/