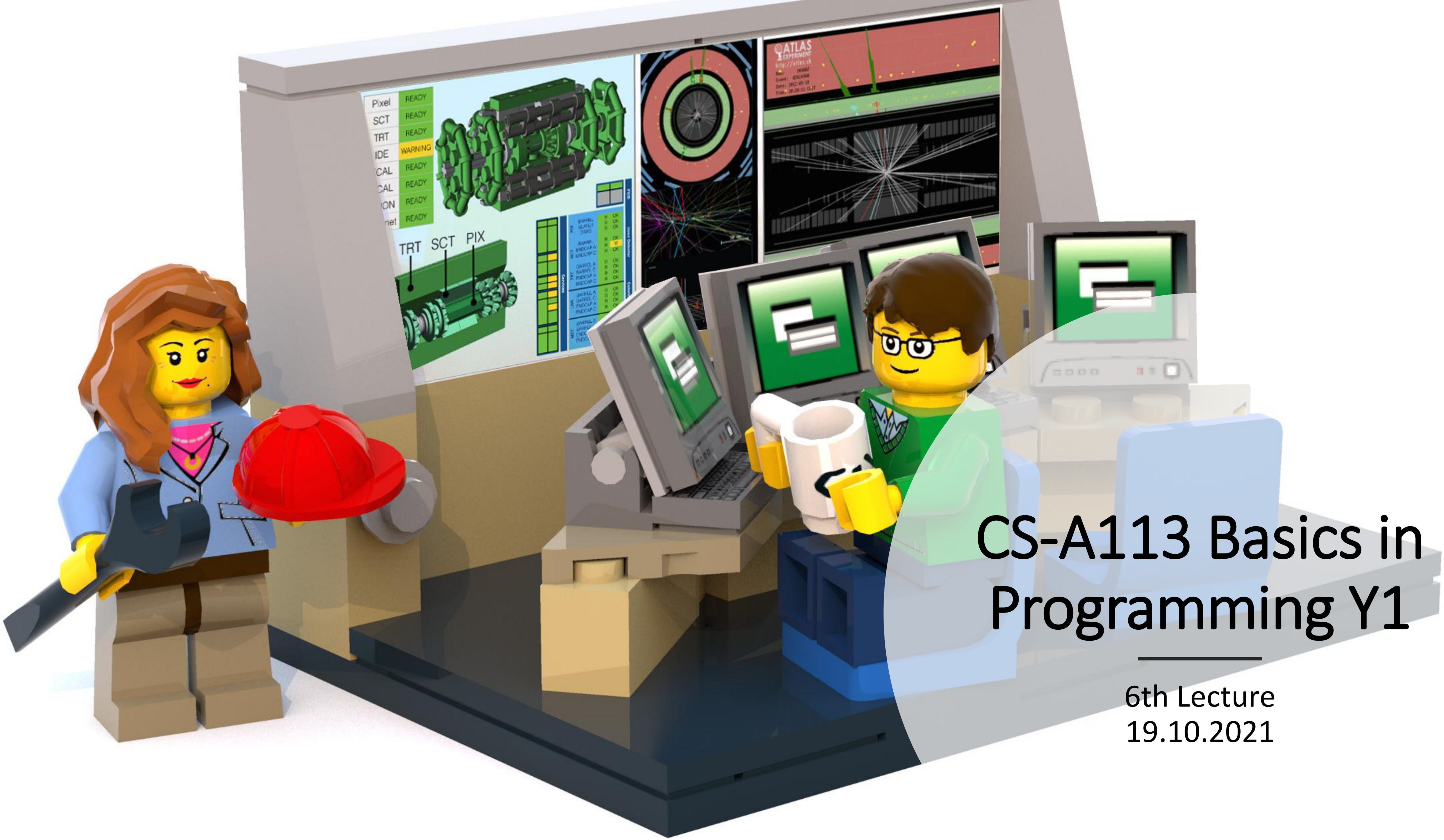


CS-A113 Basics in Programming Y1

6th Lecture
19.10.2021

The Lecture

- **Join with Video** – Makes my life nicer!
- Feel free to open your microphone and ask questions
- Feel free to write questions into the chat
- We will record the sessions and put it unlisted on youtube.

Course Information

- Mandatory Questionnaire:
  Deadline: Sunday 31st. 23:59
  Feedback to help me improve the lecture
  - what do you like about it
  - what do you dislike about it
- You will get personal coding feedback for your code you submitted for round 4 until Thursday
- Special Session in Planning for week 44 or 45

Until Thursday 28.10.2021 11:00

- Find Team Members
- Meet (preferably on-site) to decide on a Team-Name
- Post your Team-Name onto our slack Channel with team member names #team-announcement
- Each team member must react on Slack to their own announcement (✓)

Course Information

Team Up and you will get 50 points extra! Deadline for announcement of group (3-5) and name Thursday 28th of October

No lecture next week (week 43)

No exercise sessions next week

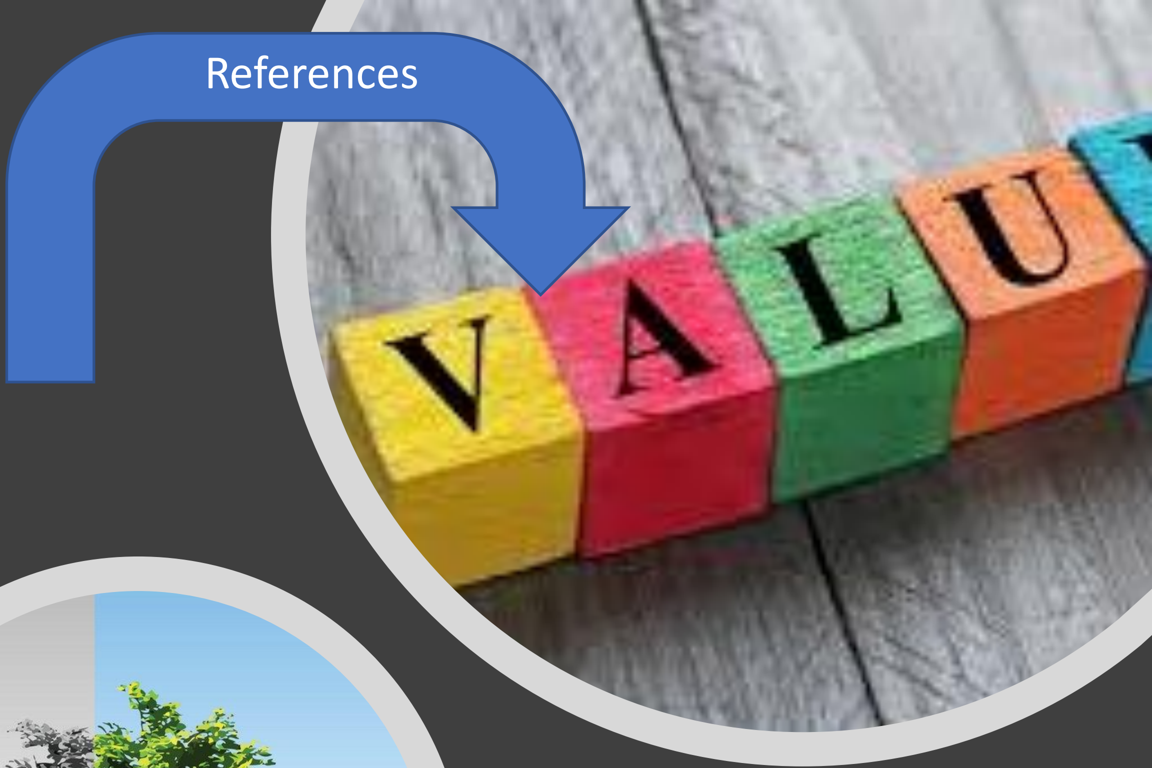
No exercise deadline next week

Don’t forget the deadline for this week 21.10;)
Don’t forget the deadline for your team 28.10;)
Don’t forget the deadline for the questionnaire 31.10 ;)

Interactions Today:

Go to:
http://presemo.aalto.fi/csa1113

Topics Today



References

Dictionairies

Binary Search

Values and References

Whats the difference?

Go to:

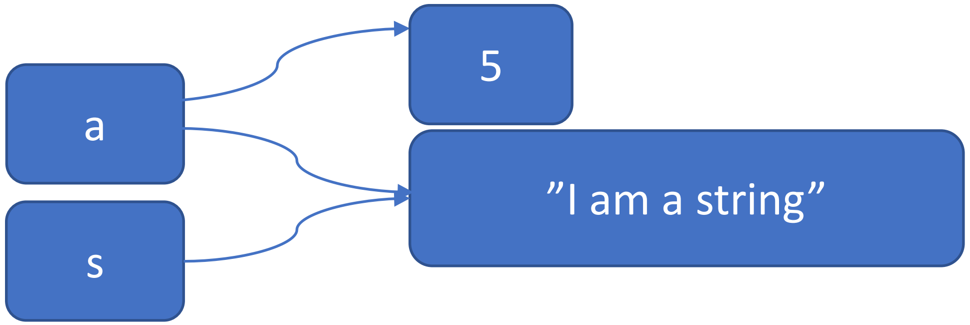
http://presemo.aalto.fi/a1113

In Python basically everything is a reference

Thats why we can make crazy things like this (please don´t ;))

Phyton

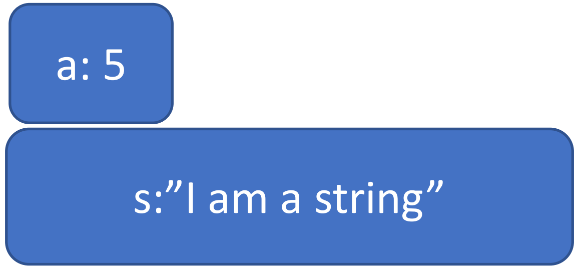
```
a = 5
s = ”I am a string”
a = s
```



Java

```
int a = 5;
string s = ”I am a string";

a = s
```

⚡ Compiler Error



Examples 1

```
def avgTemperature2():
    nofDays = 5
    temperatures = [ ]
    for i in range(nofDays):
        temperatures.append(int(input(”Enter the temperature”)))
    sum = 0
    for myTemp in temperatures:
        sum += myTemp
    avgTemp = sum/nofDays
    print(”Your average temperature is”, avgTemp)
```

nofDays = value or reference?

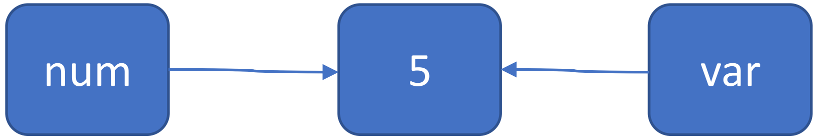
5 = value or reference?

temperatures = value or reference?

Examples 2

```
def change_number(var):
    print("Value before in change_number: ", var)
    var = 10
    print("Value after in change_number", var)


def main():
    num = 5
    print("Value before in main: ", num)
    change_number(num)
    print("Value after in main", num)
```

```
Value before in main: 5
Value before in change_number: 5
Value after in change_number: 10
Value after in main: 5
```


num

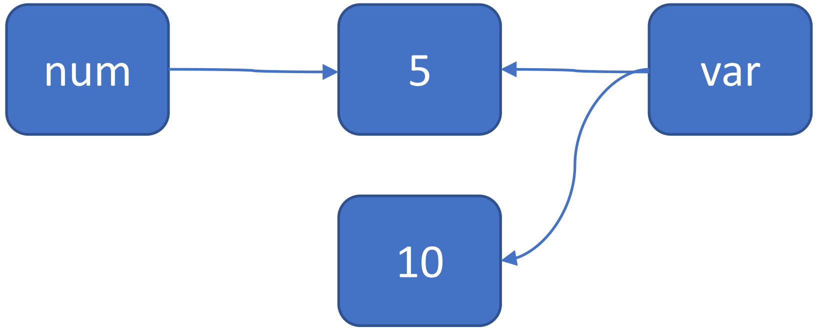
5

var

Examples 2

```
def change_number(var):
    print("Value before in change_number: ", var)
    var = 10
    print("Value after in change_number", var)


def main():
    num = 5
    print("Value before in main: ", num)
    change_number(num)
    print("Value after in main", num)
```

```
Value before in main: 5
Value before in change_number: 5
Value after in change_number: 10
Value after in main: 5
```



Passing parameter by Value

Examples 3

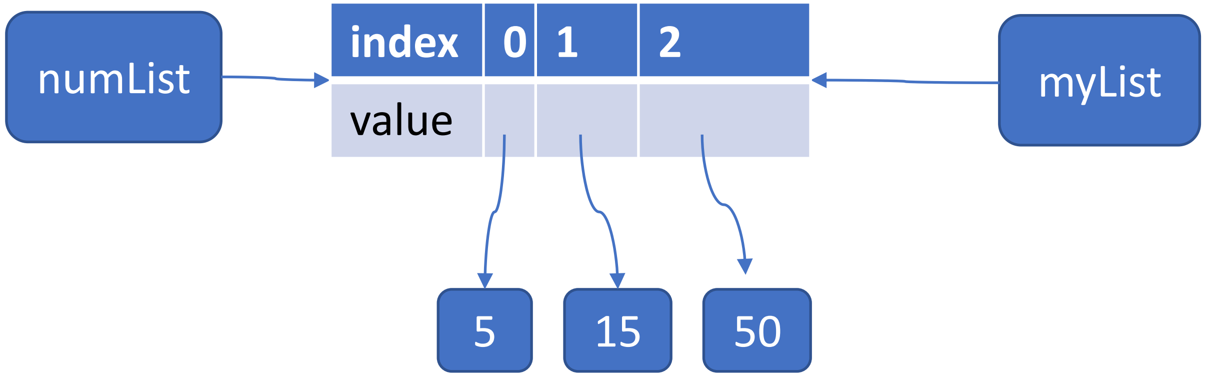
```
def  change_element(myList):

    print(“List before in change_element: ", myList)
    myList[1] = 10
    print(“List after in change_element", myList)


def  main():

    numList = [5,15,50]
    print(“List before in main: ", numList)
    change_element(numList)
    print("Value after in main", numList)
```

```
List before in main: [5,15,50]

List before in change_element: [5,15,50]

List after in change_element: [5,10,50]

List after in main: [5,10,50]
```

Examples 3

```
def  change_element(myList):

    print(“List before in change_element: ", myList)
    myList[1] = 10
    print(“List after in change_element", myList)


def  main():

    numList = [5,15,50]
    print(“List before in main: ", numList)
    change_element(numList)
    print("Value after in main", numList)
```

```
List before in main: [5,15,50]

List before in change_element: [5,15,50]

List after in change_element: [5,10,50]

List after in main: [5,10,50]
```

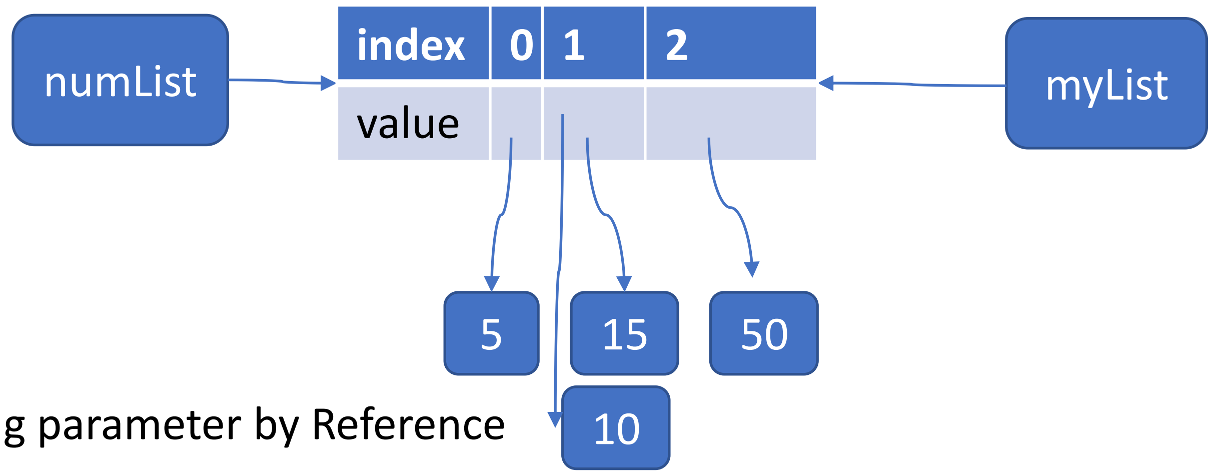


index	0	1	2
value			

Examples 3

```
def change_element(myList):

    print(“List before in change_element: ", myList)
    myList[1] = 10
    print(“List after in change_element", myList)


def main():

    numList = [5,15,50]
    print(“List before in main: ", numList)
    change_element(numList)
    print("Value after in main", numList)
```

```
List before in main: [5,15,50]

List before in change_element: [5,15,50]

List after in change_element: [5,10,50]

List after in main: [5,10,50]
```



index	0	1	2
value			

Passing parameter by Reference

Examples 4

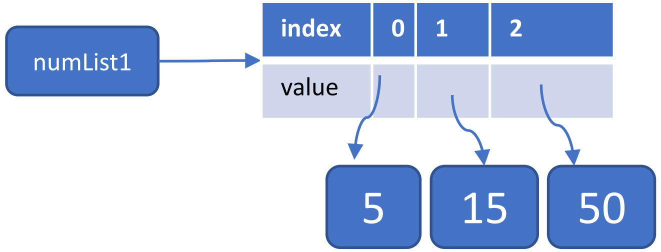
```
def  change_element(myList):

    print(“List before in change_element: ", myList)
    myList[1] = 10
    print(“List after in change_element", myList)


def  main():

    numList1  = [5,15,50]
    print(“List before in main: ", numList1)
    numList2 = numList1.copy()
    change_element(numList2)
    print(“List1 after in main", numList1)
    print (“List2 after in main", numList2)
```

```
List before in main: [5,15,50]

List before in change_element: [5,15,50]

List after in change_element: [5,10,50]

List1 after in main: [5,15,50]

List2 after in main: [5,10,50]
```

Examples 4

```
def change_element(myList):

    print(“List before in change_element: ", myList)
    myList[1] = 10
    (“List after in change_element", myList)


def main():

    numList1 = [5,15,50]
    print(“List before in main: ", numList1)
    numList2 = numList1.copy()
    change_element(numList2)
    print(“List1 after in main", numList1)
    print (“List2 after in main", numList2)
```

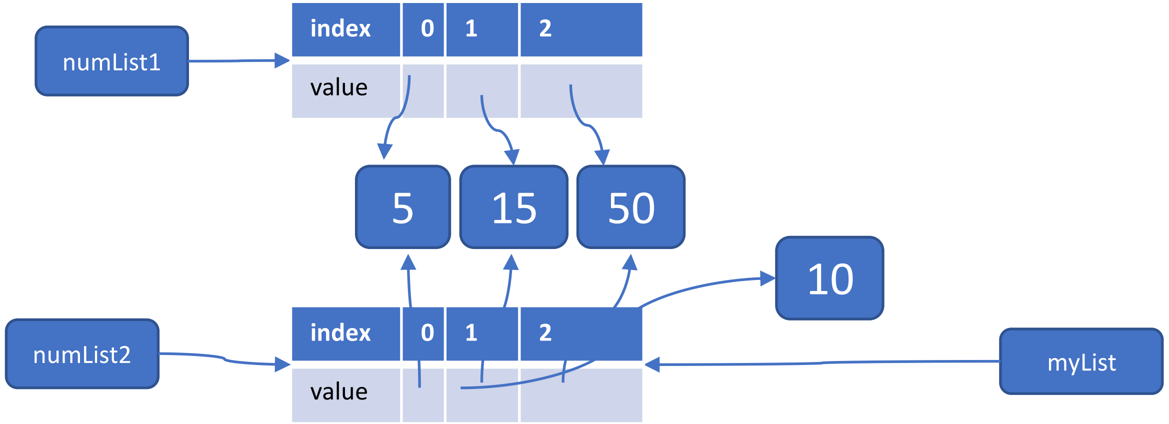
```
List before in main: [5,15,50]

List before in change_element: [5,15,50]

List after in change_element: [5,10,50]

List1 after in main: [5,15,50]

List2 after in main: [5,10,50]
```



Examples 4

```
def change_element(myList):
    print(“List before in change_element: ", myList)
    myList[1] = 10
    (“List after in change_element", myList)


def main():
    numList1 = [5,15,50]
    print(“List before in main: ", numList1)
    numList2 = numList1.copy()
    change_element(numList2)
    print(“List1 after in main", numList1)
    print (“List2 after in main", numList2)
```

```
List before in main: [5,15,50]

List before in change_element: [5,15,50]

List after in change_element: [5,10,50]

List1 after in main: [5,15,50]

List2 after in main: [5,10,50]
```



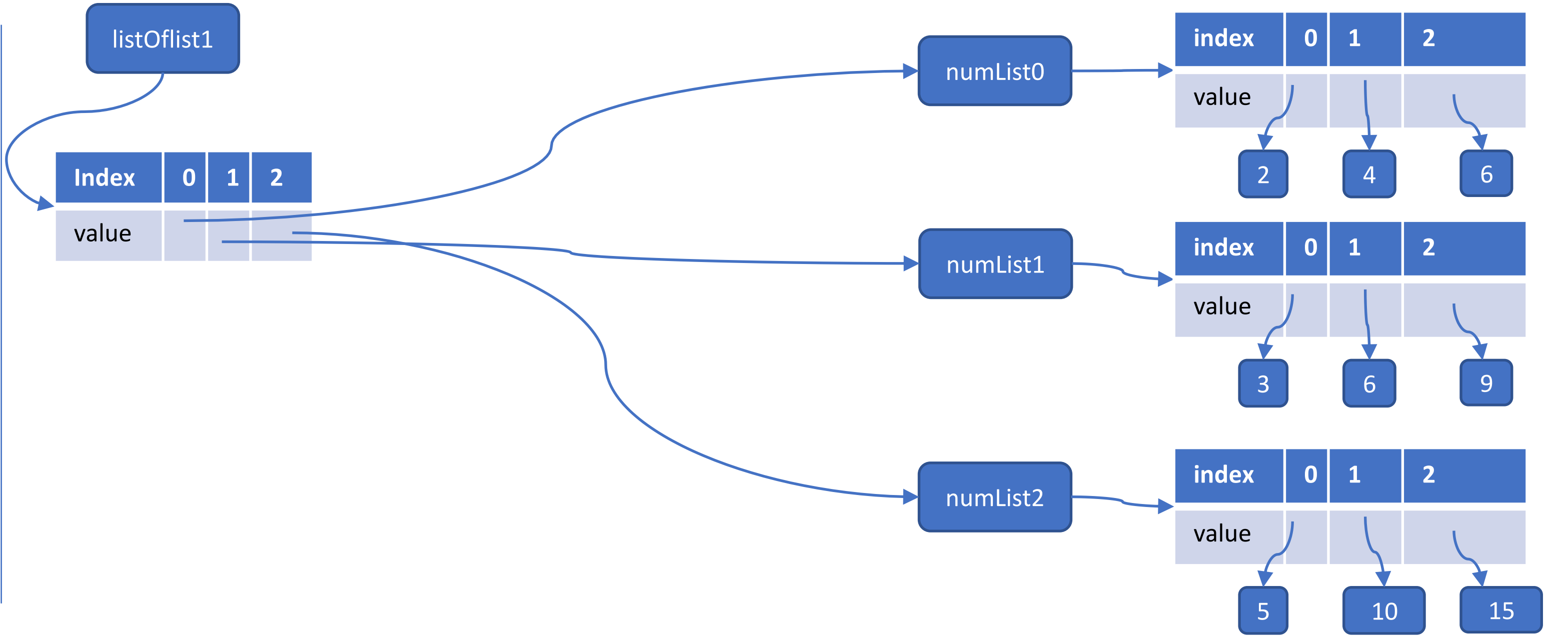
Examples 5

```
def change_element1(myList):
    myList[1][1] = 20

def main():
    numList0 = [2,4,6]
    numList1 = [3,6,9]
    numList2 = [5,10,15]

    listOfList1 = [numList1,numList2,numList3]
    listOfList2 = listOfList1.copy()

    change_element1(listOfList2)
    print(“listOfList1 after in main", numList1)
    print (“listOfList2 after in main", numList2)
```

1. listOfList1 after in main: [[20,4,6] , [3,6,9], [5,10,15]]
   listOfList2 after in main: [[20,4,6] , [3,6,9], [5,10,15]]
2. listOfList1 after in main: [[2,4,6] , [3,6,9], [5,10,15]]
   listOfList2 after in main: [[20,4,6] , [3,6,9], [5,10,15]]
3. listOfList1 after in main: [[2,4,6] , [3,20,9], [5,10,15]]
   listOfList2 after in main: [[2,4,6] , [3,20,9], [5,10,15]]
4. listOfList1 after in main: [[2,4,6] , [3,6,9], [5,10,15]]
   listOfList1 after in main: [[2,4,6] , [3,20,9], [5,10,15]]
5. listOfList1 after in main: [[2,4,6] , [3,6,9], [5,10,15]]
   listOfList1 after in main: [[2,4,6] , [3,6,9], [5,10,15]]

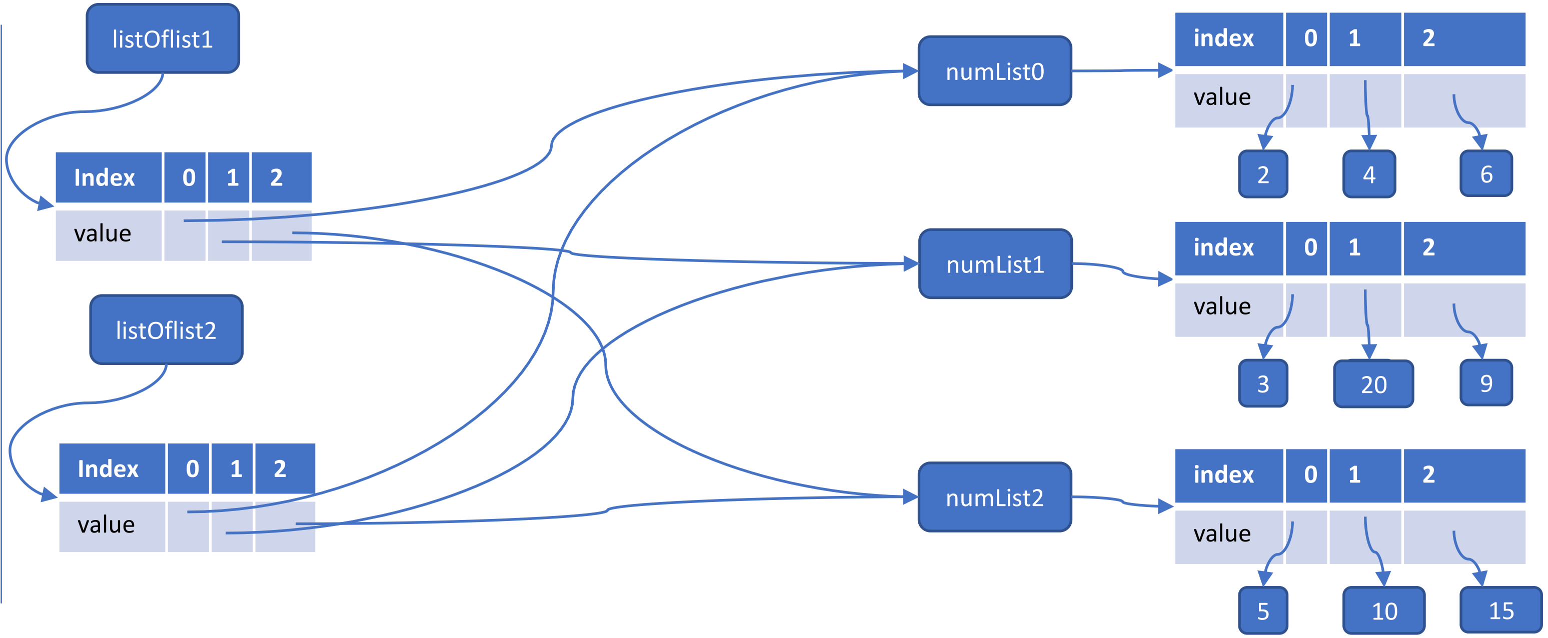
Examples 5

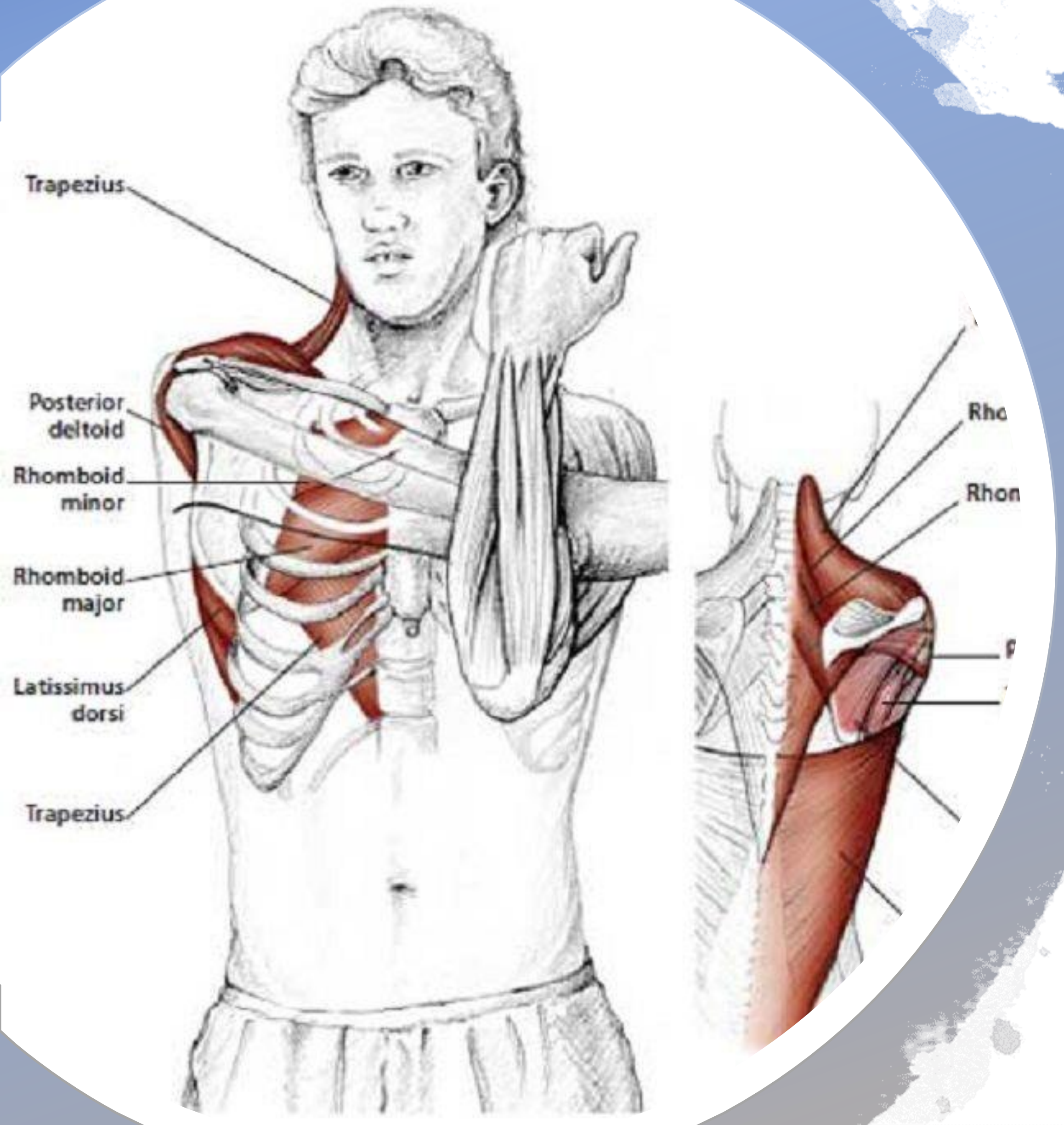
```
def change_element1(myList):
    myList[1][1] = 20

def main():
    numList0 = [2,4,6]
    numList1 = [3,6,9]
    numList2 = [5,10,15]

    listOfList1 = [numList1,numList2,numList3]
    listOfList2 = listOfList1.copy()

    change_element1(listOfList2)
    print(“listOfList1 after in main", numList1)
    print (“listOfList2 after in main", numList2)
```



Examples 5

```
def change_element1(myList):
    myList[1][1] = 20

def main():
    numList0 = [2,4,6]
    numList1 = [3,6,9]
    numList2 = [5,10,15]

    listOfList1 = [numList1,numList2,numList3]
    listOfList2 = listOfList1.copy()

    change_element1(listOfList2)
    print(“listOfList1 after in main", numList1)
    print (“listOfList2 after in main", numList2)
```





Break: Move your Shoulders

Phonebook
How?

Lists? Ordered by number?
Takes a lot of space
Ordered according to names?
Takes long to search!

Go to:

http://presemo.aalto.fi/a1113

Solution: Dictionaries

set of *key: value* pairs,
with the requirement that the keys are unique
(within one dictionary).
Keys must be immutable types (e.g., strings/numbers)

How to Start a Dictionary

```
myPhonebook = {}
myPhonebook[”Alex”] = 123
myPhonebook[”Babette”] = 365
myPhonebook[”Carl”] = 874
```

or

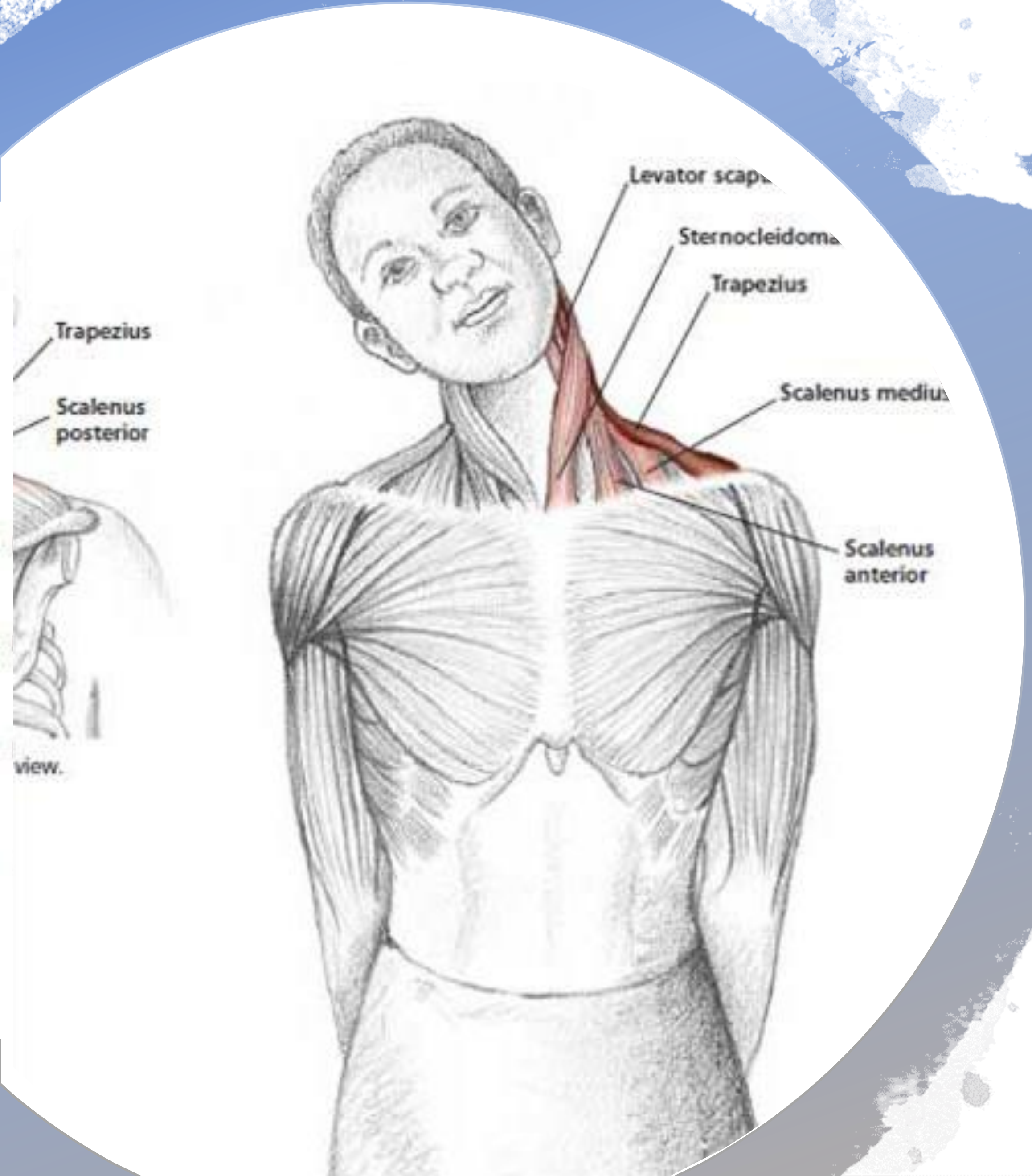
```
myPhonebook = {”Alex”:123, ”Babette”:365,”Carl”:874}
```

How to Use a Dictionary

```
myPhonebook = {”Alex”:123, ”Babette”:365,”Carl”:874}

# Need to look up Alex´ number
AlexNr = myPhonebook[”Alex”]

# Alex has a new number 234 → need to update it
myPhonebook[”Alex”] =234

# Alex has no number anymore
del myPhonebook [”Alex”]

AlexNr = myPhonebook[”Alex”]
→ keyError ”Alex”

# Is Alex still in my phonebook?
if ”Alex” in myPhonebook:
   print(myPhonebook [”Alex”] )
```

How Loop Through a Dictionary

```
myPhonebook = {”Alex”:123, ”Babette”:365,”Carl”:874}

for name in myPhonebook:
   print(name, myPhonebook[name] )

for name, phonenumber in myPhonebook.items():
   print(name,phonenumber)

myKeys = list(myPhonebook)
```



Break:
Move your Neck!

Binary Search

Finally an Algorithm ☺

How do you search?

How do you search?

I give you a list and I tell you, that it is ordered. Where is the number 80?

Hint:
The list contains values from 6 to 9798

Index	0	1	2	3	4	5	6	7	8	9	10	11	12
value													

How do you search?

I give you a list and I tell you, that it is ordered. Where is the number 80?

Hint:
The list contains values from 6 to 9798

Index	0	1	2	3	4	5	6	7	8	9	10	11	12
value	6												9798

How do you search?

I give you a list and I tell you, that it is ordered. Where is the number 80?
Hint:
The list contains values from 6 to 9798

Index	0	1	2	3	4	5	6	7	8	9	10	11	12
value	6												

Index	0	1	2	3	4	5	6	7	8	9	10	11	12
value	6	7											

.
.
.

Index	0	1	2	3	4	5	6	7	8	9	10	11	12
value	6	7	19	25	50	55	60	79					

Index	0	1	2	3	4	5	6	7	8	9	10	11	12
value	6	7	19	25	50	55	60	79	79.5				

Index	0	1	2	3	4	5	6	7	8	9	10	11	12
value	6	7	19	25	50	55	60	79	79.5	79.75	79.9	80	9798

Can we search faster?

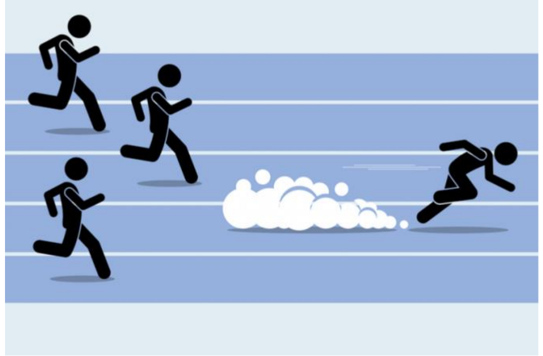
I give you a list and I tell you, that it is ordered. Where is the numb

Hint:

The list contains values from 6 to 9798

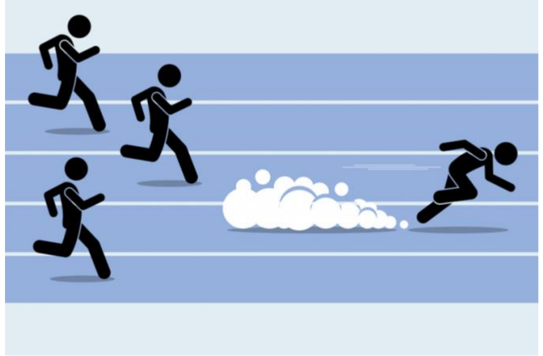
What is here?

Index	0	1	2	3	4	5	6	7	8	9	10	11	12
value													



Can we search faster?

60<80?



I give you a list and I tell you, that it is ordered. Where is the numb

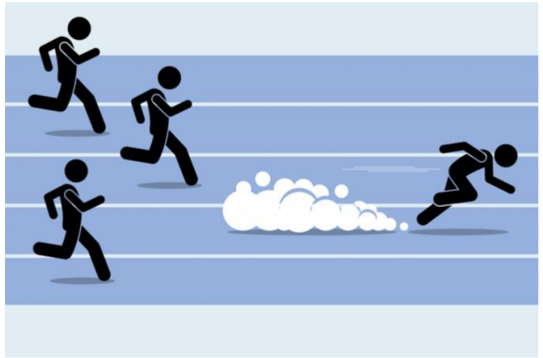
Hint:

The list contains values from 6 to 9798

Index	0	1	2	3	4	5	6	7	8	9	10	11	12
value							60						

Index	0	1	2	3	4	5	6	7	8	9	10	11	12
value							60						

Continue search only in this part



Can we search faster?

I give you a list and I tell you, that it is ordered. Where is the number 80?
Hint:
The list contains values from 6 to 9798

Index	0	1	2	3	4	5	6	7	8	9	10	11	12
value							60						

Index	0	1	2	3	4	5	6	7	8	9			12
value							60						

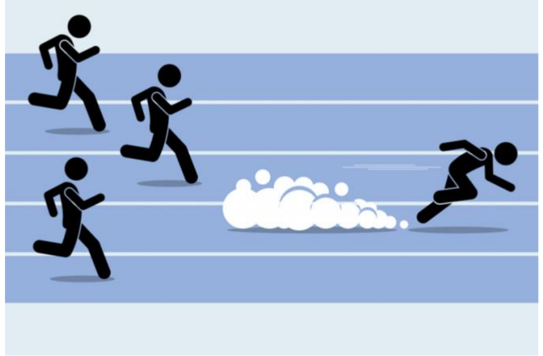
79.75<80?

Whats here?

Index	0	1	2	3	4	5	6	7	8	9	10	11	1
value							60			79.75	Continue		

Index	0	1	2	3	4	5	6	7	8	9	10	11	12
value							60			79.75			

Can we search faster?



I give you a list and I tell you, that it is ordered. Where is the number 80?
Hint:
The list contains values from 6 to 9798

Index	0	1	2	3	4	5	6	7	8	9	10	11	12
value							60						

Index	0	1	2	3	4	5	6	7	8	9	10	11	12
value							60						

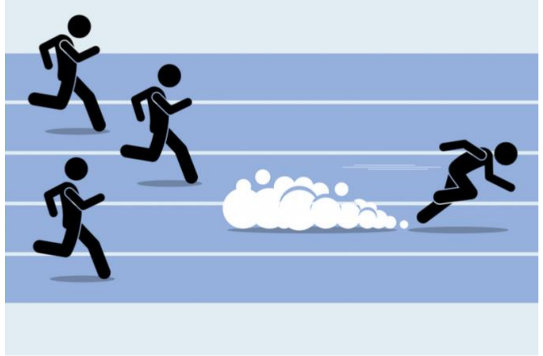
Index	0	1	2	3	4	5	6	7	8	9	10	11	12
value							60			79.75			

Index	0	1	2	3	4	5	6	7	8	9	10	11	
value							60			79.75			

Index	0	1	2	3	4	5	6	7	8	9	10	11	12
value							60			79.75		80	

Found it!

Can we search faster?



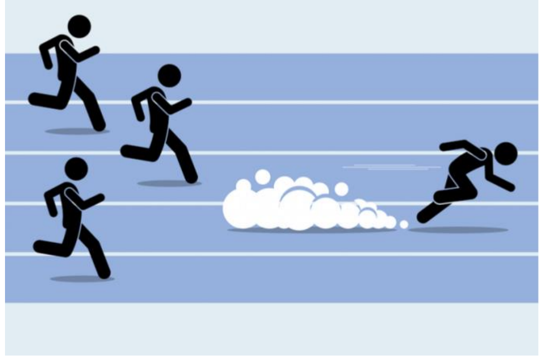
We have a sequential Algorithme (just go trough the list 1 by 1)
And we have this other Algorithm.

How can we compare them? How do we know which one is better?

As it is dependent on the input (if 80 is on the first spot we find it immediately in the sequential Algorithm) we need to find a better comparison than just 1 run.

In Computer Science we often compare the worst-case scenario:
When does the Algorithm terminate with the worst possible input for THIS Algorithm?

Can we search faster?



Why are we faster?

We halve the search area in each step instead of making it one smaller

How fast are we in the worst case if n= len(list)

Instead of $n = len(list)$ we have $\log_2 n$ steps

n	$\log_2 n$
2	1
4	2
8	3
16	4
32	5
64	6
128	7

n	$\log_2 n$
256	8
512	9
1024	10
2048	11
4096	12
8192	13
16384	14

n	$\log_2 n$
32768	15
10^6	~20
10^9	~30
10^{12}	~40
10^{21}	~70
10^{30}	~100
…	…

"That's all Folks!"
Isberg©