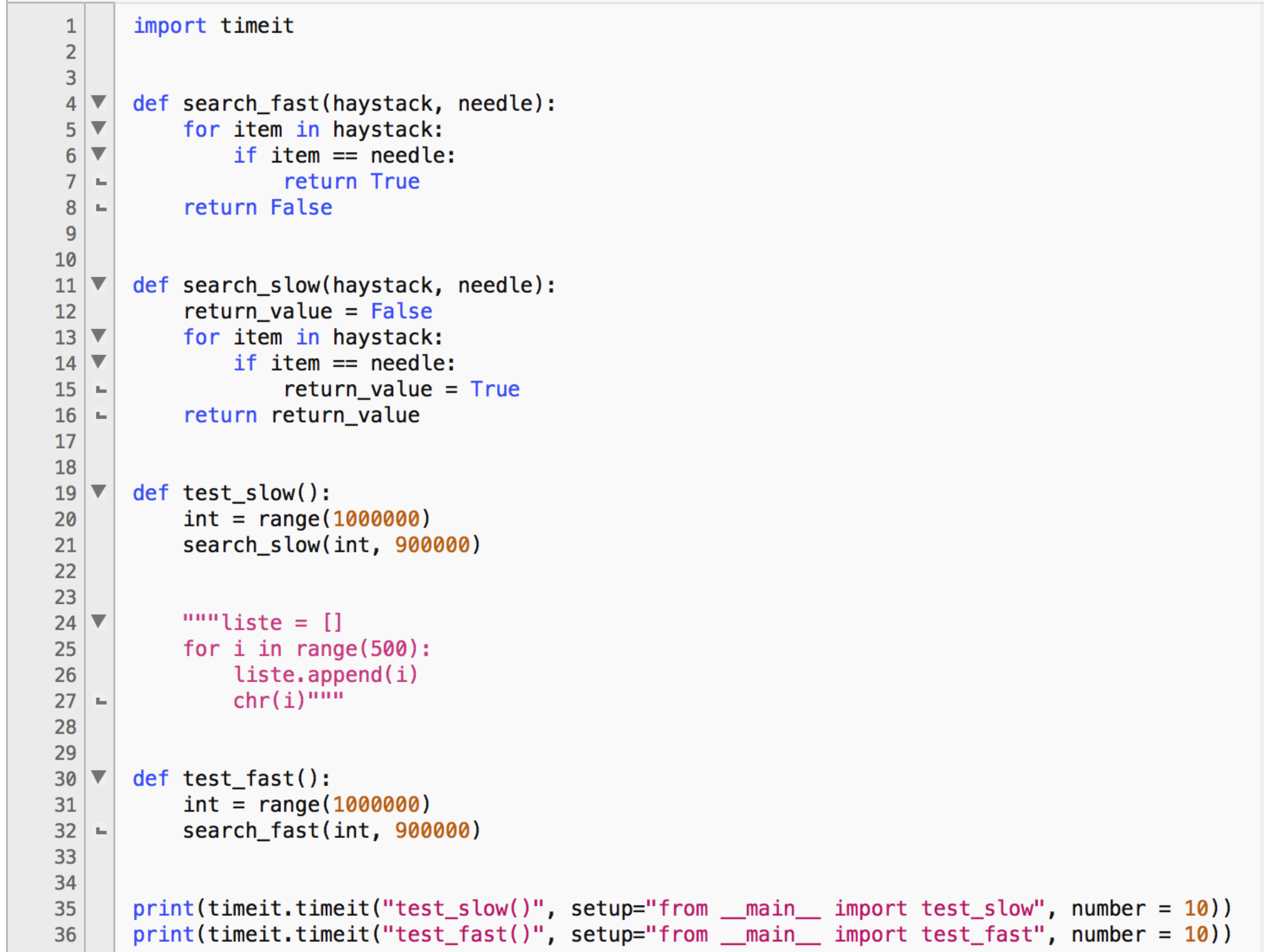
Gruppe 10 Presentasjon

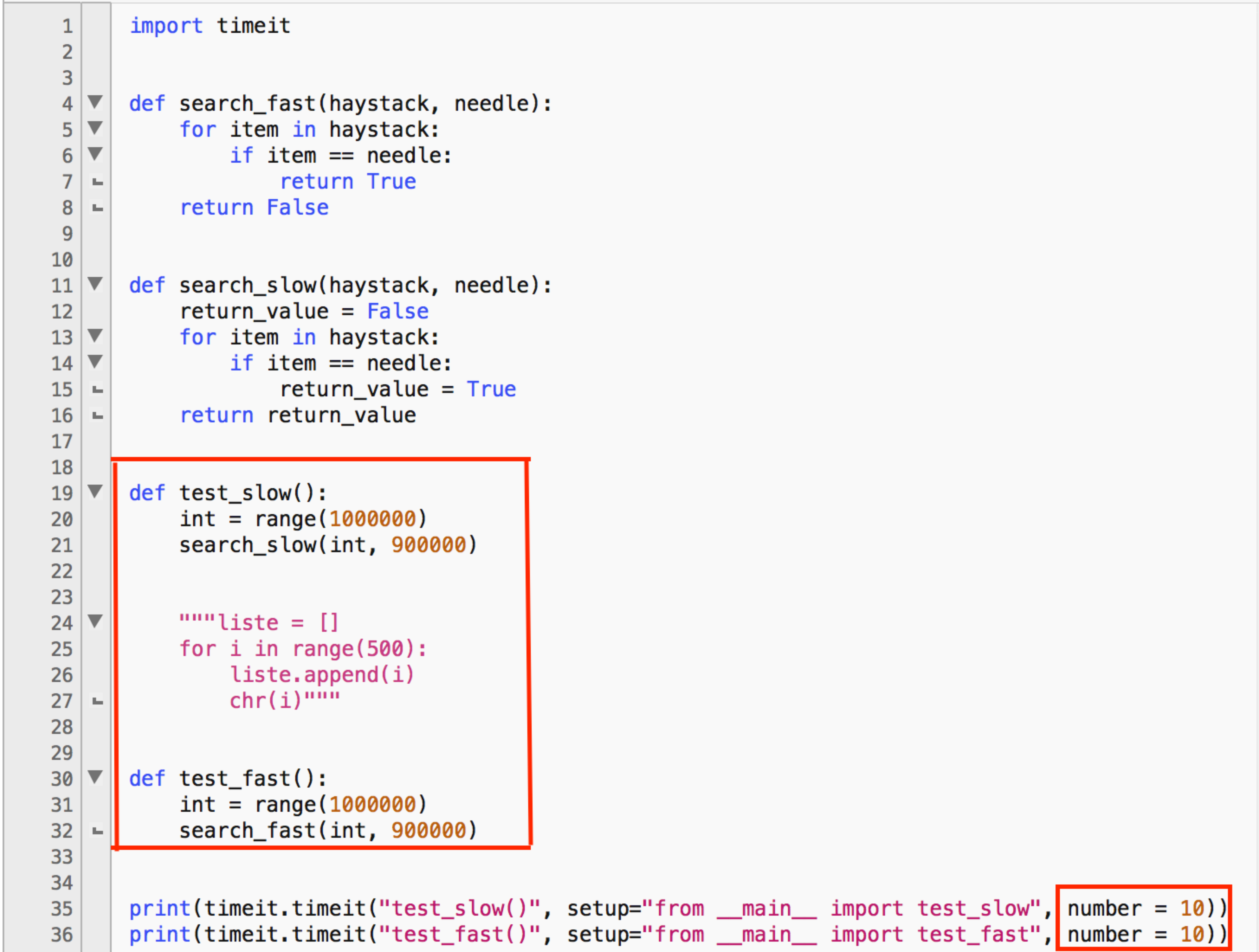
-*@æ

Sammenligning av pcer i gruppe10

Beskrivelse	Magnus	Erik	Mohammad/John/ Eirik	Per
CPU	2.3 GHz	2.5 GHz	2.7 GHz	3 GHz
RAM	4 GB	8 GB	8 GB	8 GB
L2	256 KB	256 KB	256 KB	256 KB
L3	3 MB	3 MB	3 MB	4 MB
VRAM	385 MB	1024 MB	1536 MB	1536 MB
Grafikkort	Intel HD Graphics 3000	Intel HD Graphics 4000	Intel Iris Graphics 6100	Intel HD Graphics 4000
Operativsystem	OS X El Capitan	OS X El Capitan	OS X El Capitan	OS X El Capitan
Prosessor	IntelCore i5	IntelCore i5	IntelCore i5	IntelCore i7
Pc type	MacBook pro 2011	MacBook pro 2014	MacBook pro 2015	MacBook pro 2013

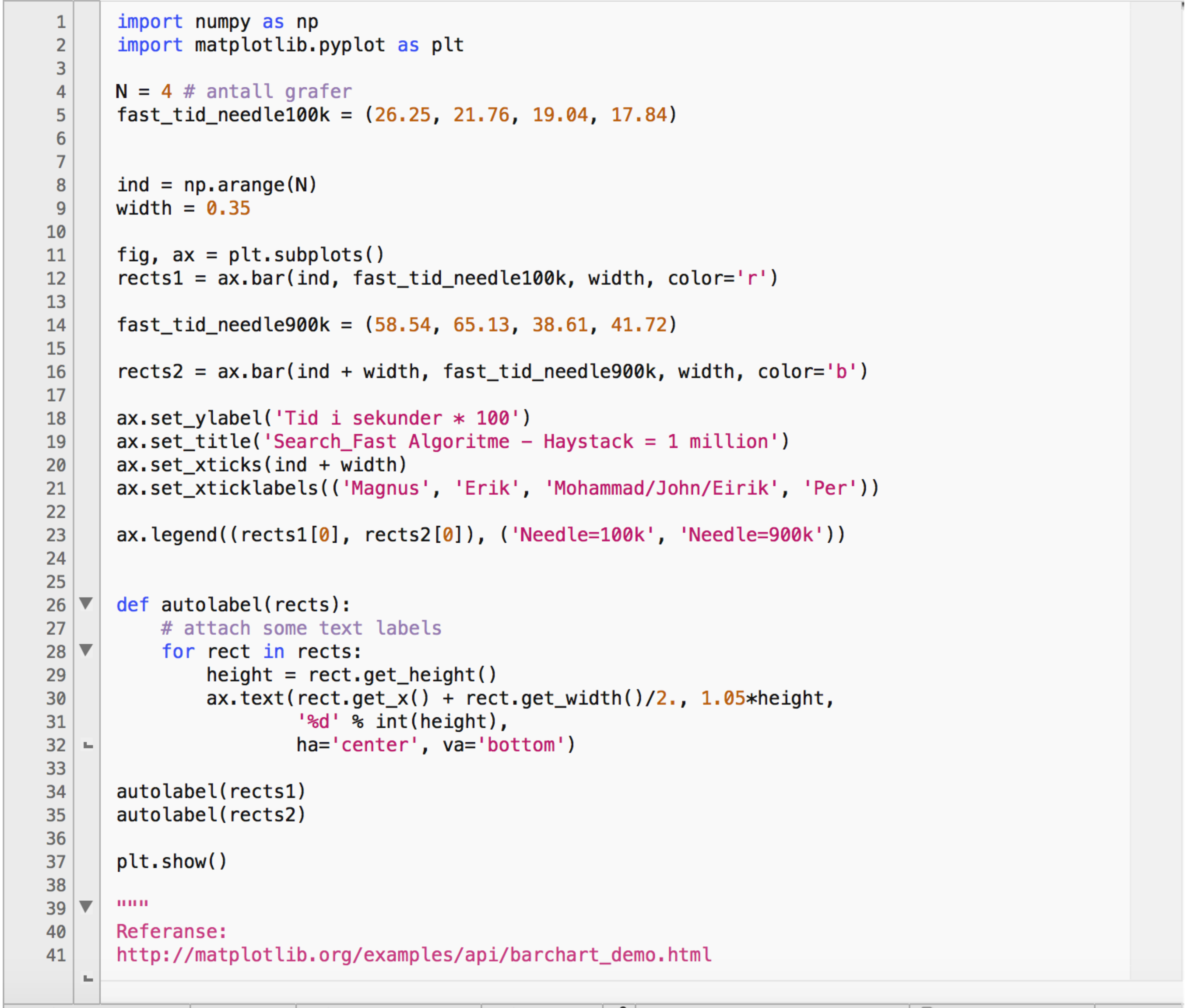


```python
import timeit


def search_fast(haystack, needle):
    for item in haystack:
        if item == needle:
            return True
    return False


def search_slow(haystack, needle):
    return_value = False
    for item in haystack:
        if item == needle:
            return_value = True
    return return_value


def test_slow():
    int = range(1000000)
    search_slow(int, 900000)


    """liste = []
    for i in range(500):
        liste.append(i)
        chr(i)"""


def test_fast():
    int = range(1000000)
    search_fast(int, 900000)


print(timeit.timeit("test_slow()", setup="from __main__ import test_slow", number = 10))
print(timeit.timeit("test_fast()", setup="from __main__ import test_fast", number = 10))
```



```python
import timeit


def search_fast(haystack, needle):
    for item in haystack:
        if item == needle:
            return True
    return False


def search_slow(haystack, needle):
    return_value = False
    for item in haystack:
        if item == needle:
            return_value = True
    return return_value


def test_slow():
    int = range(1000000)
    search_slow(int, 900000)


    """liste = []
    for i in range(500):
        liste.append(i)
        chr(i)"""


def test_fast():
    int = range(1000000)
    search_fast(int, 900000)


print(timeit.timeit("test_slow()", setup="from __main__ import test_slow", number = 10))
print(timeit.timeit("test_fast()", setup="from __main__ import test_fast", number = 10))
```

Beskrivelse	Magnus	Erik	Mohammad/John/ Eirik	Per
Slow_Test 1000 Needle = 100	0.00692105293274	0.000840902328491	0.000334978103638	0.000322818756104
Fast_Test 1000 Needle = 100	0.00692105293274	0.000183820724487	0.000190019607544	0.000103950500488
Slow_Test 1000, Needle = 900	0.00692105293274	0.000571966171265	0.000314950942993	0.000343084335327
Fast_Test 1000, Needle = 900	0.00174117088318	0.000526189804077	0.000287055969238	0.000336885452271
Slow_Test 10.000 Needle = 1000	0.00519990921021	0.00746989250183	0.00351214408875	0.00349712371826
Fast_Test 10.000 Needle = 1000	0.00393009185791	0.00197386741638	0.00105810165405	0.00100302696228
Slow_Test 10.000 Needle = 9000	0.004469871521	0.00577807426453	0.00381803512573	0.00330519676208
Fast_Test 10.000 Needle = 9000	0.00920295715332	0.00547790527344	0.00301599502563	0.00324106216431
Slow_Test 100.00 Needle = 10.000	0.0650689601898	0.0659861564636	0.0393869876862	0.0338110923767
Fast_Test 100.00 Needle = 10.000	0.0226318836212	0.0217688083649	0.0128870010376	0.0112421512604
Slow_Test 100.00 Needle = 90.000	0.0625698566437	0.0923109054565	0.042277097702	0.041796207428
Fast_Test 100.00 Needle = 90.000	0.0519819259644	0.106792926788	0.0314722061157	0.0321991443634
Slow_Test 1 000 000 Needle = 100.000	0.652861118317	0.718982934952	0.435479879379	0.446295022964
Fast_Test 1 000 000 Needle = 100.000	0.262537956238	0.217605829239	0.190463066101	0.178406000137
Slow_Test 1 000 000 Needle = 900.000	0.608834028244	0.713128089905	0.4127368927	0.439824104309
Fast_Test 1 000 000 Needle = 900.000	0.585485935211	0.651344060898	0.386194944382	0.439824104309



```python
import numpy as np
import matplotlib.pyplot as plt

N = 4 # antall grafer
fast_tid_needle100k = (26.25, 21.76, 19.04, 17.84)


ind = np.arange(N)
width = 0.35

fig, ax = plt.subplots()
rects1 = ax.bar(ind, fast_tid_needle100k, width, color='r')

fast_tid_needle900k = (58.54, 65.13, 38.61, 41.72)

rects2 = ax.bar(ind + width, fast_tid_needle900k, width, color='b')

ax.set_ylabel('Tid i sekunder * 100')
ax.set_title('Search_Fast Algoritme - Haystack = 1 million')
ax.set_xticks(ind + width)
ax.set_xticklabels(('Magnus', 'Erik', 'Mohammad/John/Eirik', 'Per'))

ax.legend((rects1[0], rects2[0]), ('Needle=100k', 'Needle=900k'))


def autolabel(rects):
    # attach some text labels
    for rect in rects:
        height = rect.get_height()
        ax.text(rect.get_x() + rect.get_width()/2., 1.05*height,
                '%d' % int(height),
                ha='center', va='bottom')

autolabel(rects1)
autolabel(rects2)

plt.show()

"""
Referanse:
http://matplotlib.org/examples/api/barchart_demo.html
```

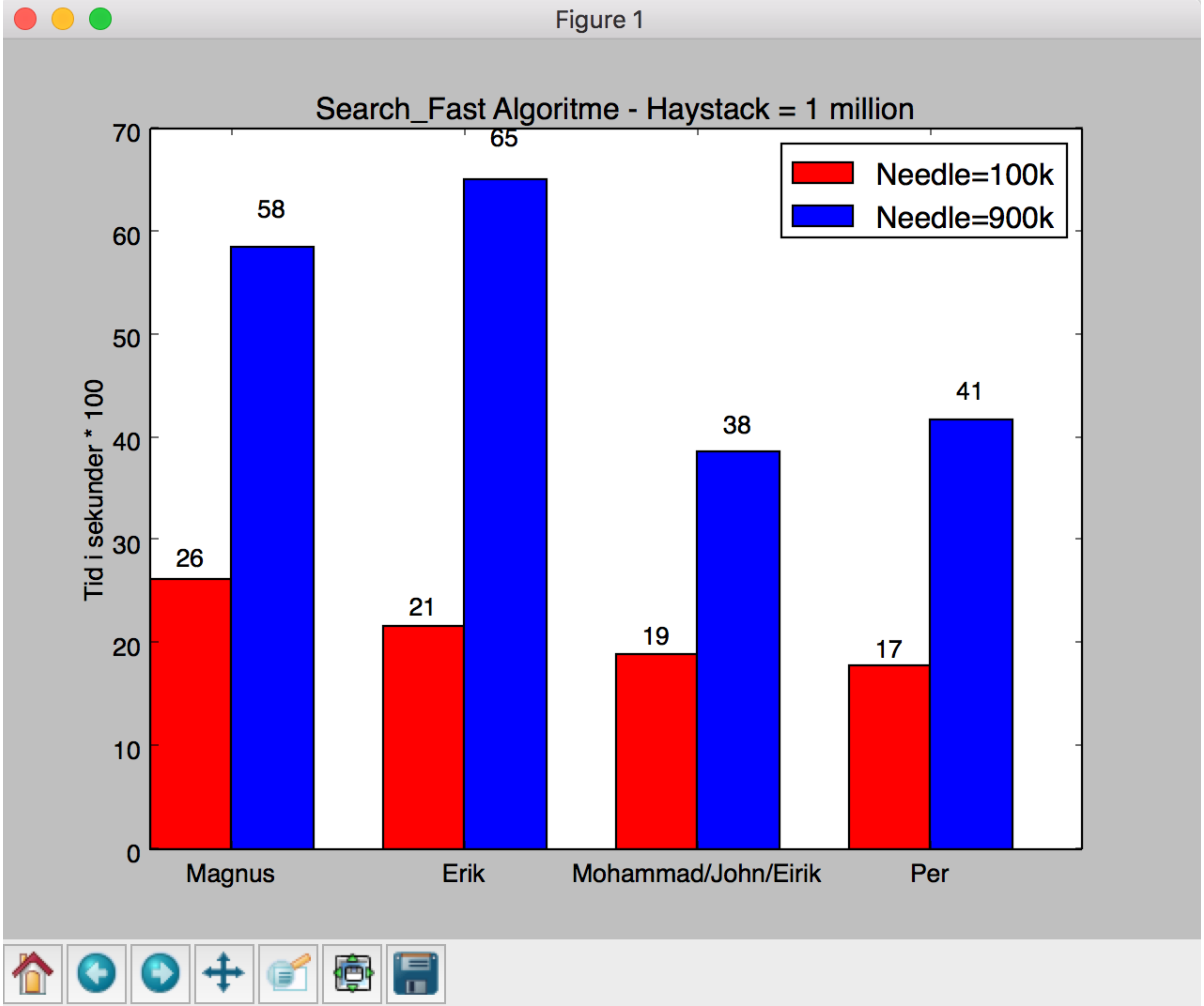


Figure 1
Search_Fast Algoritme - Haystack = 1 million
Tid i sekunder * 100
0
10
20
30
40
50
60
70
🟥 Needle=100k
🟦 Needle=900k
26
58
21
65
19
38
17
41
Magnus
Erik
Mohammad/John/Eirik
Per

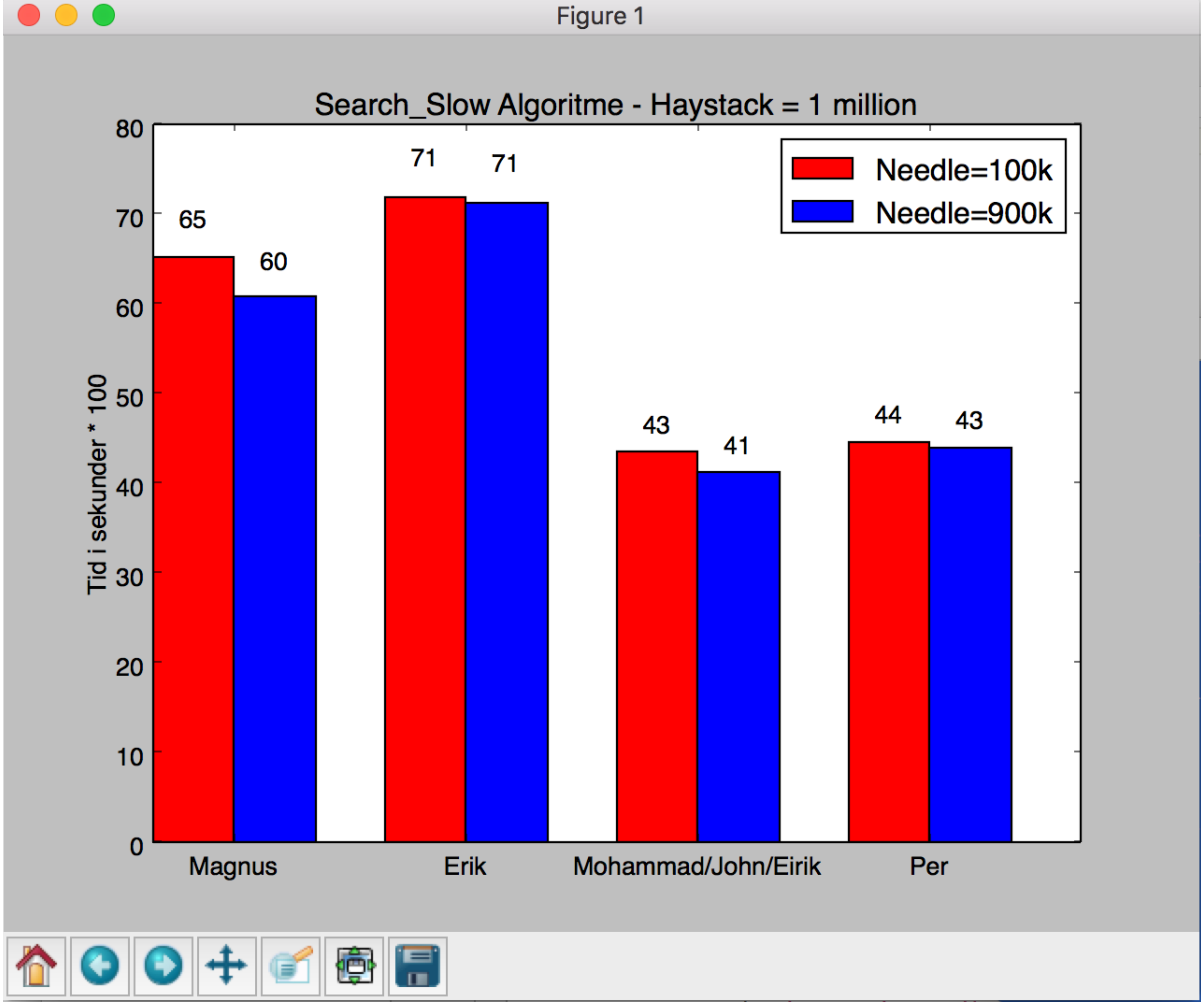
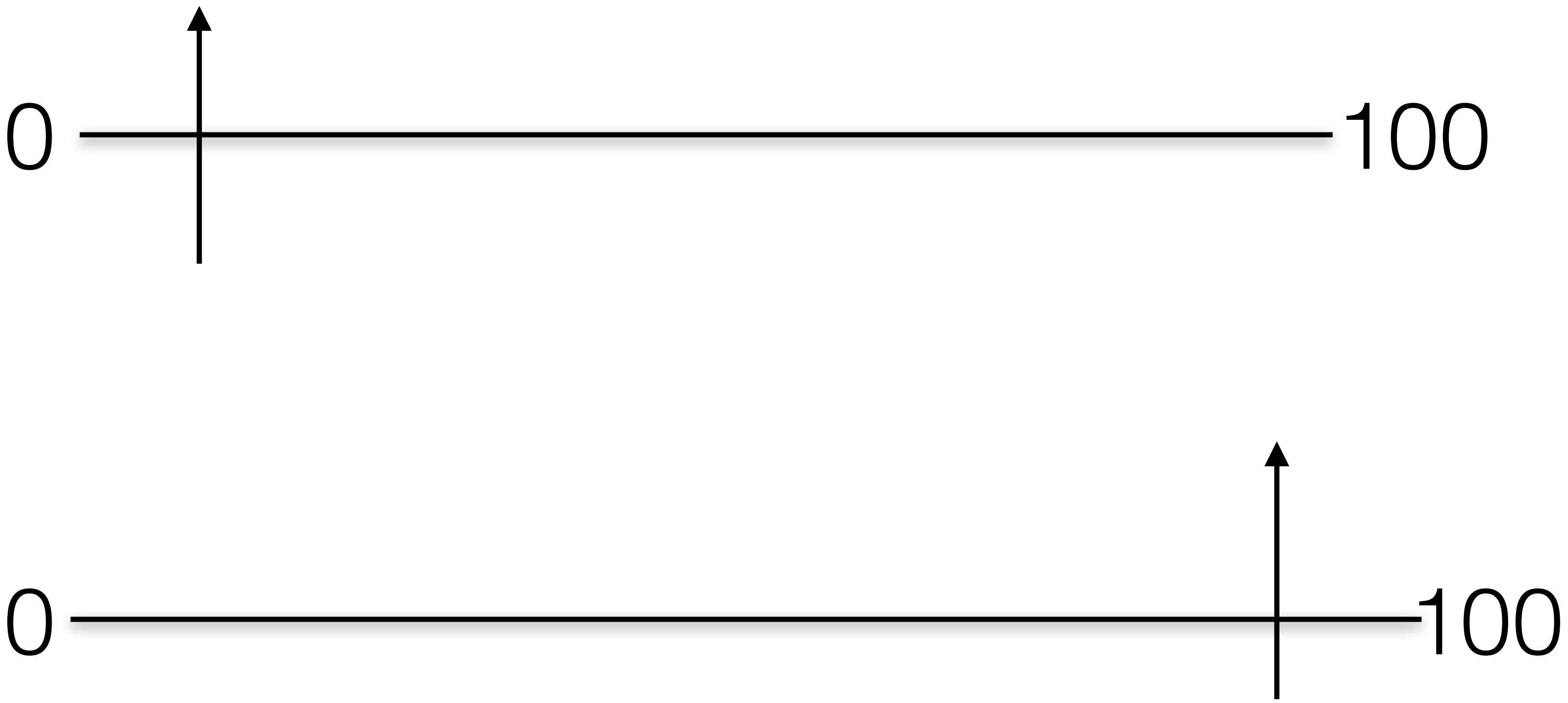


Figure 1
Search_Slow Algoritme - Haystack = 1 million
Tid i sekunder * 100
80
70
60
50
40
30
20
10
0
🟥 Needle=100k
🟦 Needle=900k
65
60
71
71
43
41
44
43
Magnus
Erik
Mohammad/John/Eirik
Per

Search_slow

Search_fast



- Gruppe 10 Medlemmer:
- Mohammad Hussain, Magnus Høvik, John Jensen, Eirik Oliversen, Erik Angell Berg, Per Dimitris