22. Searching a List

Topics:

- Linear Search
- Binary Search
- Measuring Execution Time
- The Divide and Conquer Framework

Search

Examples:

Is this song in that playlist?

Is this number in that phone book?

Is this name in that phone book?

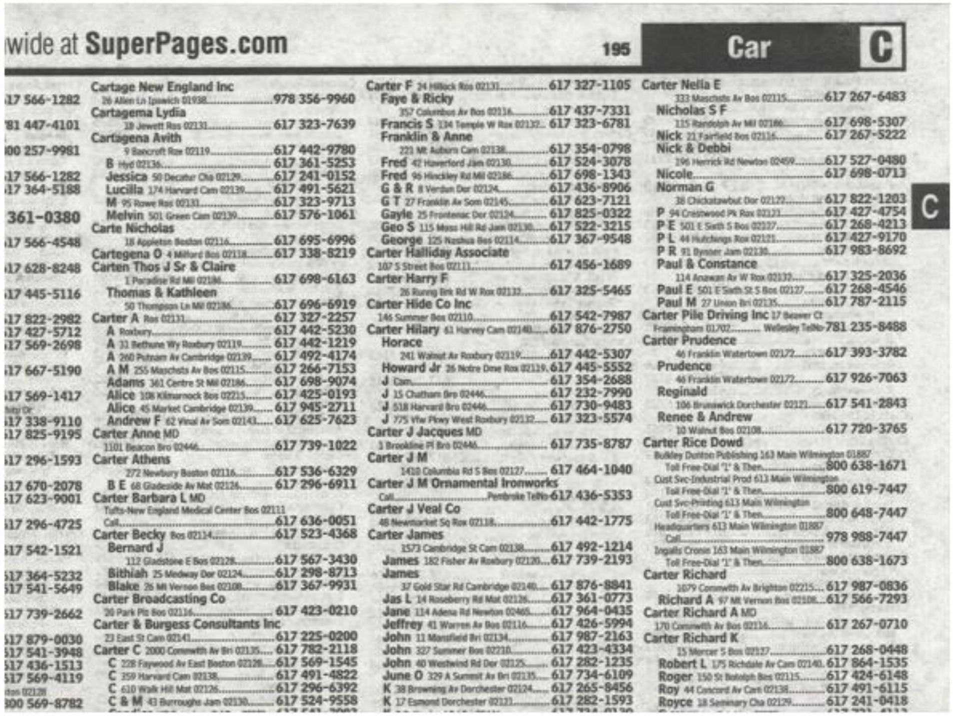
Is this fingerprint in that archive of fingerprints?

Is this photo in that yearbook?

More on Using Phone Books

The Manhatten phone book has 1,000,000+ entries.

How is it possible to locate a name by examining just a <u>tiny</u> , <u>tiny</u> fraction of those entries?



wide at SuperPages.com 195 Car C

17 566-1282
81 447-4101
00 257-9981
17 566-1282
17 364-5188
361-0380
17 566-4548
17 628-8248
17 445-5116
17 822-2982
17 427-5712
17 569-2698
17 667-5190
17 569-1417
17 338-9110
17 825-9195
17 296-1593
17 670-2078
17 623-9001
17 296-4725
17 542-1521
17 364-5232
17 541-5649
17 739-2662
17 879-0030
17 541-3948
17 436-1513
17 569-4119
ton 02128
00 569-8782

Cartage New England Inc 26 Allen Ln Ipswich 01938	978 356-9960
Cartagema Lydia 18 Jewett Ros 02131	617 323-7639
Cartagena Avith 9 Bancroft Rox 02119	617 442-9780
B Hyd 02136	617 361-5253
Jessica 50 Decatur Cha 02129	617 241-0152
Lucilla 174 Harvard Cam 02139	617 491-5621
M 95 Rowe Ros 02131	617 323-9713
Melvin 501 Green Cam 02139	617 576-1061
Carte Nicholas 18 Appleton Boston 02116	617 695-6996
Cartegena O 4 Milford Bos 02118	617 338-8219
Carten Thos J Sr & Claire 1 Paradise Rd Mil 02186	617 698-6163
Thomas & Kathleen 50 Thompson Ln Mil 02186	617 696-6919
Carter A Ros 02131	617 327-2257
A Roxbury	617 442-5230
A 31 Bethune Wy Roxbury 02119	617 442-1219
A 260 Putnam Av Cambridge 02139	617 492-4174
A M 255 Maschsts Av Bos 02115	617 266-7153
Adams 361 Centre St Mil 02186	617 698-9074
Alice 106 Kilmarnock Bos 02215	617 425-0193
Alice 45 Market Cambridge 02139	617 945-2711
Andrew F 62 Vinal Av Som 02143	617 625-7623
Carter Anne MD 1101 Beacon Bro 02446	617 739-1022
Carter Athens 272 Newbury Boston 02116	617 536-6329
B E 68 Gladeside Av Mat 02126	617 296-6911
Carter Barbara L MD Tufts-New England Medical Center Bos 02111 Call	617 636-0051
Carter Becky Bos 02114	617 523-4368
Bernard J 112 Gladstone E Bos 02128	617 567-3430
Bithiah 25 Medway Dor 02124	617 298-8713
Blake 26 Mt Vernon Bos 02108	617 367-9931
Carter Broadcasting Co 20 Park Plz Bos 02116	617 423-0210
Carter & Burgess Consultants Inc 23 East St Cam 02141	617 225-0200
Carter C 2000 Commwlth Av Bri 02135	617 782-2118
C 228 Faywood Av East Boston 02128	617 569-1545
C 359 Harvard Cam 02138	617 491-4822
C 610 Walk Hill Mat 02126	617 296-6392
C & M 43 Burroughs Jam 02130	617 524-9558

Carter F 24 Hillock Ros 02131	617 327-1105
Faye & Ricky 357 Columbus Av Bos 02116	617 437-7331
Francis S 134 Temple W Rox 02132	617 323-6781
Franklin & Anne 221 Mt Auburn Cam 02138	617 354-0798
Fred 47 Haverford Jam 02130	617 524-3078
Fred 96 Hinckley Rd Mil 02186	617 698-1343
G & R 8 Verdun Dor 02124	617 436-8906
G T 27 Franklin Av Som 02145	617 623-7121
Gayle 25 Frontenac Dor 02124	617 825-0322
Geo S 115 Moss Hill Rd Jam 02130	617 522-3215
George 125 Nashua Bos 02114	617 367-9548
Carter Halliday Associate 107 S Street Bos 02111	617 456-1689
Carter Harry F 26 Running Brk Rd W Rox 02132	617 325-5465
Carter Hide Co Inc 146 Summer Bos 02110	617 542-7987
Carter Hilary 61 Harvey Cam 02140	617 876-2750
Horace 241 Walnut Av Roxbury 02119	617 442-5307
Howard Jr 26 Notre Dme Rox 02119	617 445-5552
J Cam	617 354-2688
J 15 Chatham Bro 02446	617 232-7990
J 518 Harvard Bro 02446	617 730-9483
J 775 Vfw Pkwy West Roxbury 02132	617 323-5574
Carter J Jacques MD 1 Brookline Pl Bro 02446	617 735-8787
Carter J M 1410 Columbia Rd S Bos 02127	617 464-1040
Carter J M Ornamental Ironworks Call Pembroke TelNo	617 436-5353
Carter J Veal Co 48 Newmarket Sq Rox 02118	617 442-1775
Carter James 1573 Cambridge St Cam 02138	617 492-1214
James 182 Fisher Av Roxbury 02120	617 739-2193
James 37 Gold Star Rd Cambridge 02140	617 876-8841
Jas L 14 Roseberry Rd Mat 02126	617 361-0773
Jane 114 Adena Rd Newton 02465	617 964-0435
Jeffrey 41 Warren Av Bos 02116	617 426-5994
John 11 Mansfield Bri 02134	617 987-2163
John 327 Summer Bos 02210	617 423-4334
John 40 Westwind Rd Dor 02125	617 282-1235
June O 329 A Summit Av Bri 02135	617 734-6109
K 38 Browning Av Dorchester 02124	617 265-8456
K 17 Esmond Dorchester 02121	617 282-1593

Carter Nella E 333 Maschsts Av Bos 02115	617 267-6483
Nicholas S F 115 Randolph Av Mil 02186	617 698-5307
Nick 21 Fairfield Bos 02116	617 267-5222
Nick & Debbi 196 Herrick Rd Newton 02459	617 527-0480
Nicole	617 698-0713
Norman G 38 Chickatawbut Dor 02122	617 822-1203
P 94 Crestwood Pk Rox 02121	617 427-4754
P E 501 E Sixth S Bos 02127	617 268-4213
P L 44 Hutchings Rox 02121	617 427-9170
P R 91 Bynner Jam 02130	617 983-8692
Paul & Constance 114 Anawan Av W Rox 02132	617 325-2036
Paul E 501 E Sixth St S Bos 02127	617 268-4546
Paul M 27 Union Bri 02135	617 787-2115
Carter Pile Driving Inc 17 Beaver Ct Framingham 01702 Wellesley TelNo	781 235-8488
Carter Prudence 46 Franklin Watertown 02172	617 393-3782
Prudence 46 Franklin Watertown 02172	617 926-7063
Reginald 106 Brunswick Dorchester 02121	617 541-2843
Renee & Andrew 10 Walnut Bos 02108	617 720-3765
Carter Rice Dowd	
Bulkley Dunton Publishing 163 Main Wilmington 01887 Toll Free-Dial "1" & Then	800 638-1671
Cust Svc-Industrial Prod 613 Main Wilmington Toll Free-Dial "1" & Then	800 619-7447
Cust Svc-Printing 613 Main Wilmington Toll Free-Dial "1" & Then	800 648-7447
Headquarters 613 Main Wilmington 01887 Call	978 988-7447
Ingalls Cronin 163 Main Wilmington 01887 Toll Free-Dial "1" & Then	800 638-1673
Carter Richard 1079 Commwlth Av Brighton 02215	617 987-0836
Richard A 97 Mt Vernon Bos 02108	617 566-7293
Carter Richard A MD 170 Commwlth Av Bos 02116	617 267-0710
Carter Richard K 15 Mercer S Bos 02127	617 268-0448
Robert L 175 Richdale Av Cam 02140	617 864-1535
Roger 150 St Botolph Bos 02115	617 424-6148
Roy 44 Concord Av Cam 02138	617 491-6115
Royce 18 Seminary Cha 02129	617 241-0418

C

There must be a great search algorithm behind the scenes.

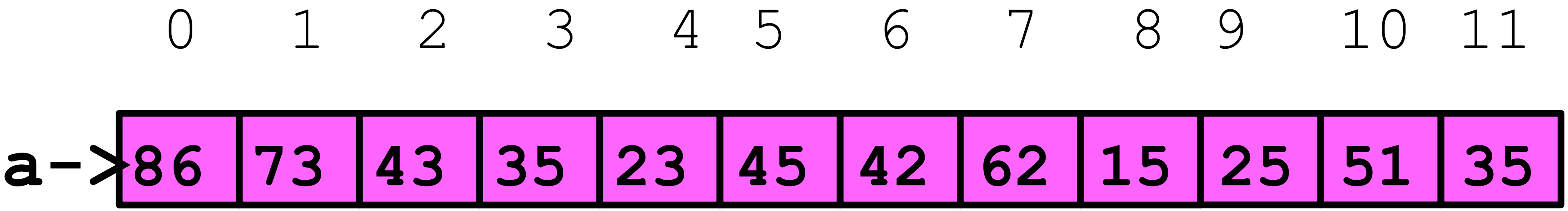
Linear Search

LinSearch: The Spec

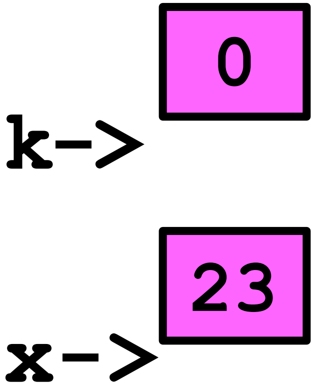
```
def LinSearch(x,a):
    """ Returns an int k with the
    property that a[k]==x is True.
    If no such k exists, then
    k==-1.

    PreC: a is a nonempty list of
    ints and x is an int.
    """
```

Could also apply the same ideas for searching a list of strings.

Linear Search



0	1	2	3	4	5	6	7	8	9	10	11
86	73	43	35	23	45	42	62	15	25	51	35



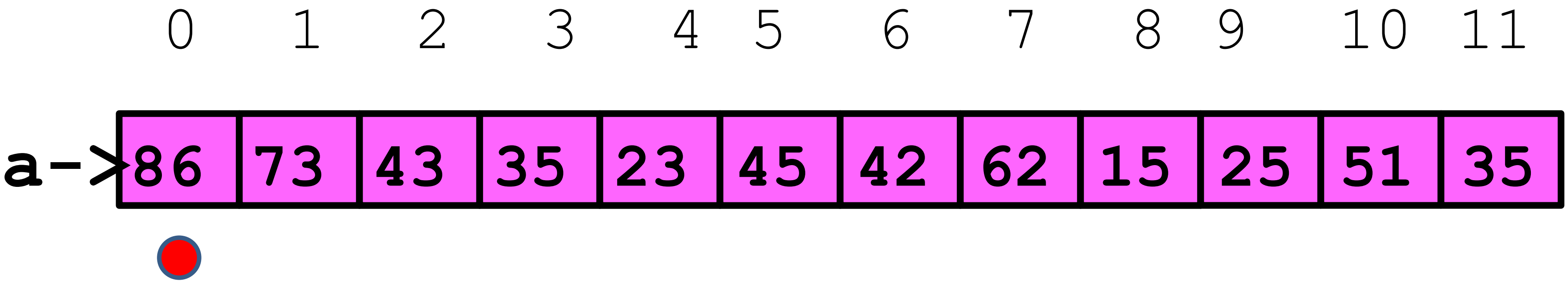
k-> 0

x-> 23

```
def  LinSearch(x,a):
    for k in range(len(a)):
        if x == a[k]:
            return k
    return -1
```

Walk down the list looking for a match

Linear Search



	0	1	2	3	4	5	6	7	8	9	10	11
a->	86	73	43	35	23	45	42	62	15	25	51	35

k-> 0

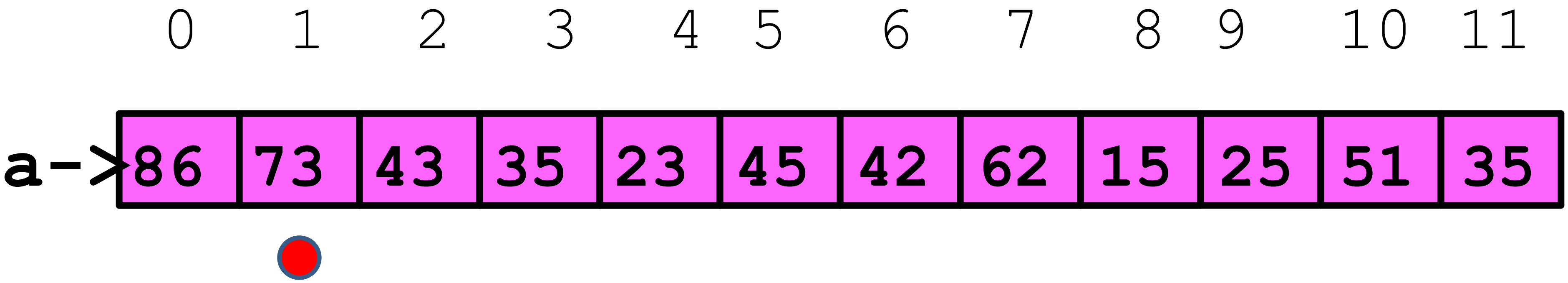
x-> 23

```
def  LinSearch(x,a):
    for k in range(len(a)):
        if x == a[k]:
            return k
    return -1
```

Nope

Walk down the list looking for a match

Linear Search



0	1	2	3	4	5	6	7	8	9	10	11
86	73	43	35	23	45	42	62	15	25	51	35

k-> 1

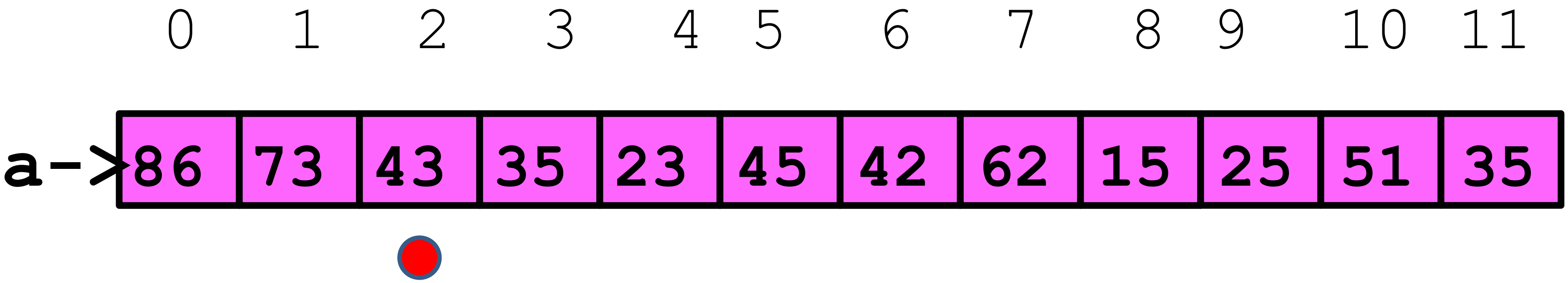
x-> 23

```
def  LinSearch(x,a):
    for k in range(len(a)):
        if x == a[k]:
            return k
    return -1
```

Nope

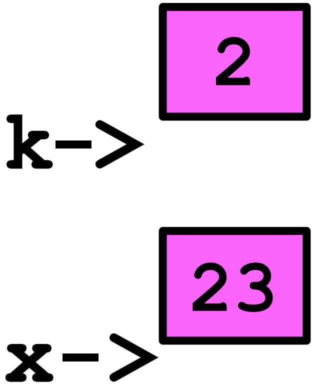
Walk down the list looking for a match

Linear Search



a->

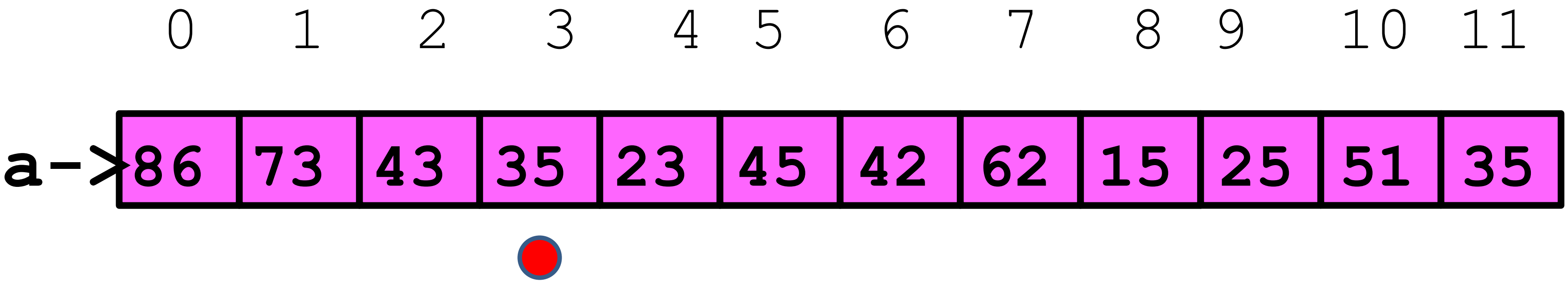
0	1	2	3	4	5	6	7	8	9	10	11
86	73	43	35	23	45	42	62	15	25	51	35



```
def  LinSearch(x,a):
   for k in range(len(a)):
       if x == a[k]:  Nope
          return k
   return -1
```

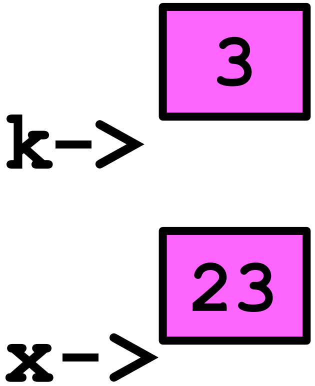
Walk down the list looking for a match

Linear Search



a->

0	1	2	3	4	5	6	7	8	9	10	11
86	73	43	35	23	45	42	62	15	25	51	35



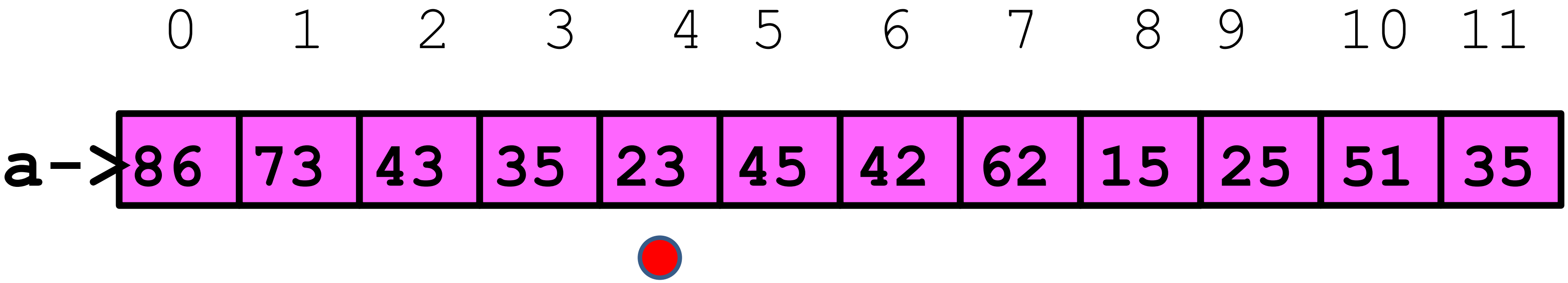
k-> 3

x-> 23

```
def  LinSearch(x,a):
    for k in range(len(a)):
        if x == a[k]:  Nope
            return k
    return -1
```

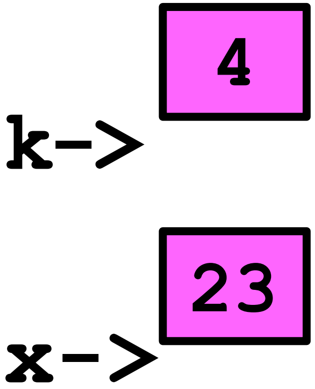
Walk down the list looking for a match

Linear Search



a->

0	1	2	3	4	5	6	7	8	9	10	11
86	73	43	35	23	45	42	62	15	25	51	35



k-> 4

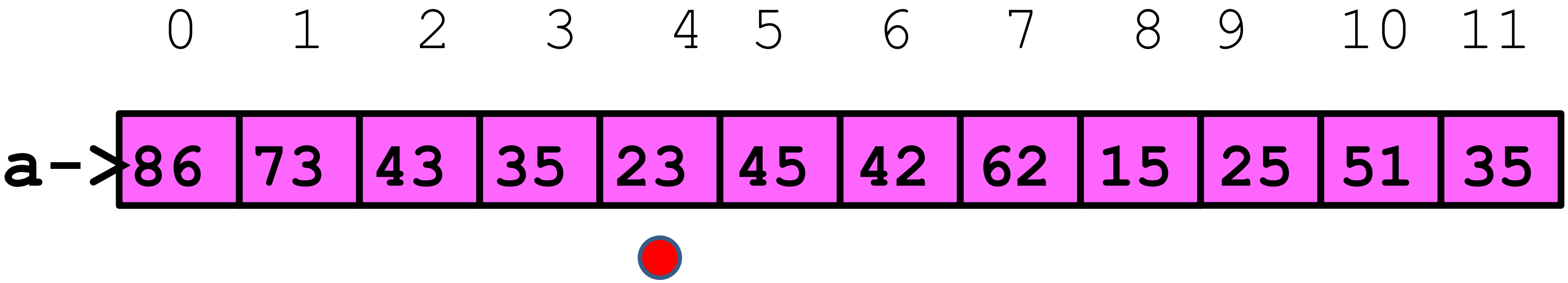
x-> 23

```
def  LinSearch(x,a):
    for k in range(len(a)):
        if x == a[k]:
            return k
    return -1
```

Yup

Walk down the list looking for a match

Linear Search



	0	1	2	3	4	5	6	7	8	9	10	11
a->	86	73	43	35	23	45	42	62	15	25	51	35

k-> 4

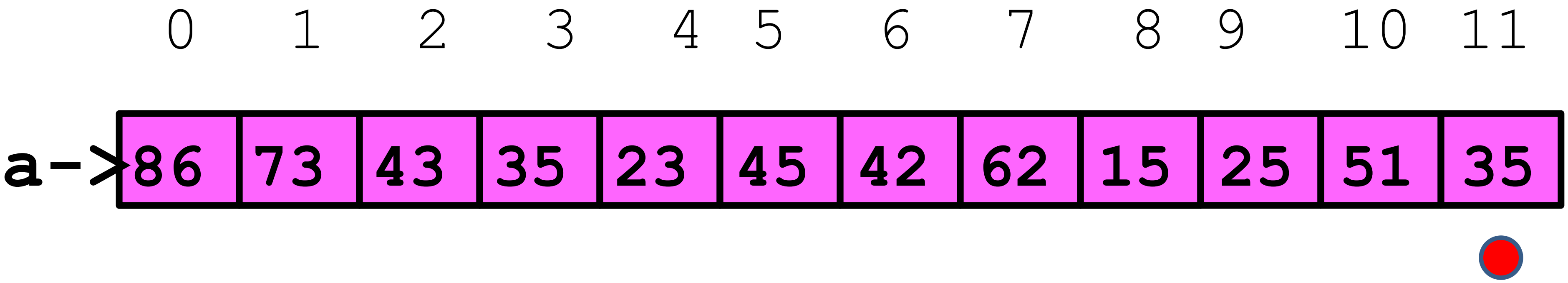
x-> 23

```
def  LinSearch(x,a):
    for k in range(len(a)):
        if x == a[k]:
            return k
    return -1
```

All done

Walk down the list looking for a match

Linear Search: No Match Case



	0	1	2	3	4	5	6	7	8	9	10	11
a->	86	73	43	35	23	45	42	62	15	25	51	35

k-> 11

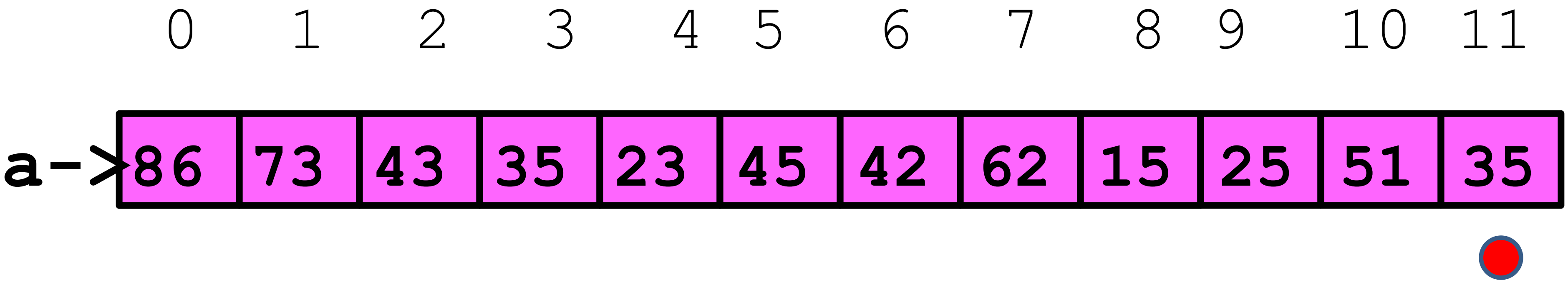
x-> 7

```
def  LinSearch(x,a):
    for k in range(len(a)):
        if x == a[k]:
            return k
    return -1
```

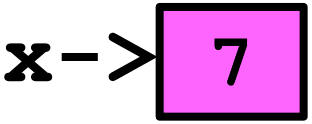
Nope

Walk down the list looking for a match

Linear Search: No Match Case



	0	1	2	3	4	5	6	7	8	9	10	11
a->	86	73	43	35	23	45	42	62	15	25	51	35
												●



x-> 7

```
def  LinSearch(x,a):
   for k in range(len(a)):
       if x == a[k]:
          return k
   return -1
```

Yup

Return -1 if no match

Linear Search: While Implementation

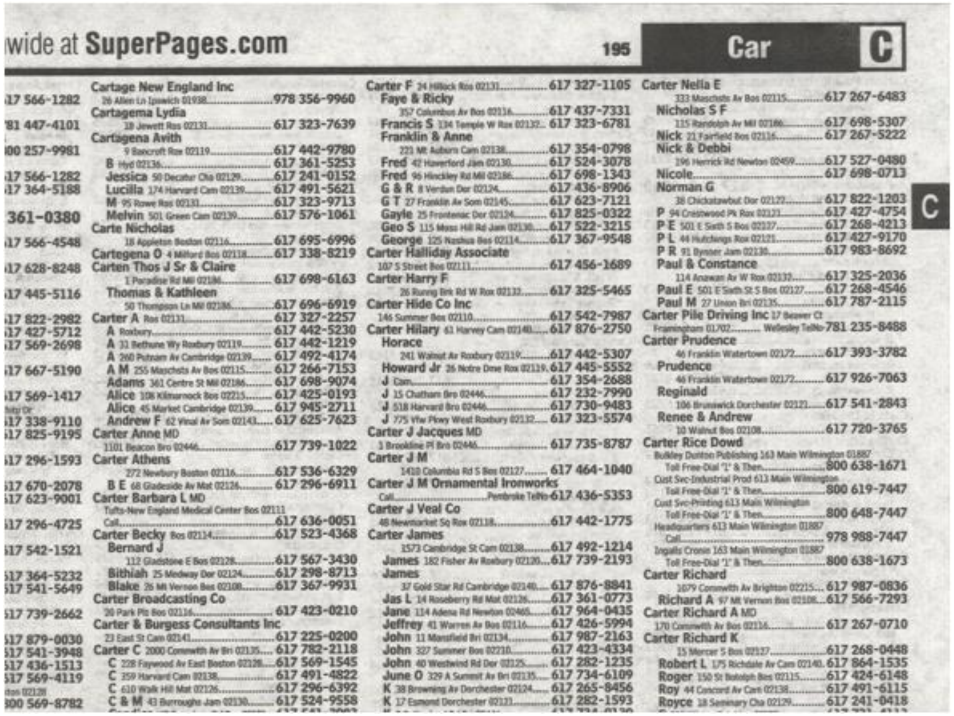
```
def  LinSearchW(x,a):
   k=0
   while k<len(a) and a[k]!=x:
       k+=1
   if k==len(a):
       return -1
   else:
       return k
```

Binary Search

Now we assume that the list
to be searched is sorted
from little to big.

```
a = [10,20,40,60,90]

a = [‘brown’,’dog’,’fox’,‘lazy’,‘quick’,’the’]
```

Back to Using Phone Books



wide at **SuperPages.com** 195 **Car** **C**

17 566-1282
81 447-4101
00 257-9981
17 566-1282
17 364-5188
361-0380
17 566-4548
17 628-8248
17 445-5116
17 822-2982
17 427-5712
17 569-2698
17 667-5190
17 569-1417
hitp Dr
17 338-9110
17 825-9195
17 296-1593
17 670-2078
17 623-9001
17 296-4725
17 542-1521
17 364-5232
17 541-5649
17 739-2662
17 879-0030
17 541-3948
17 436-1513
17 569-4119
ton 02128
00 569-8782

Cartage New England Inc
26 Allen Ln Ipswich 01938..........978 356-9960
Cartagema Lydia
18 Jewett Ros 02131..........617 323-7639
Cartagena Avith
9 Bancroft Row 02119..........617 442-9780
B Hyd 02136..........617 361-5253
Jessica 50 Decatur Cha 02129..........617 241-0152
Lucilla 174 Harvard Cam 02139..........617 491-5621
M 95 Rowe Ros 02131..........617 323-9713
Melvin 501 Green Cam 02139..........617 576-1061
Carte Nicholas
18 Appleton Boston 02116..........617 695-6996
Cartegena O 4 Milford Bos 02118..........617 338-8219
Carten Thos J Sr & Claire
1 Paradise Rd Mil 02186..........617 698-6163
Thomas & Kathleen
50 Thompson Ln Mil 02186..........617 696-6919
Carter A Ros 02131..........617 327-2257
A Roxbury..........617 442-5230
A 31 Bethune Wy Roxbury 02119..........617 442-1219
A 260 Putnam Av Cambridge 02139..........617 492-4174
A M 255 Maschsts Av Bos 02115..........617 266-7153
Adams 361 Centre St Mil 02186..........617 698-9074
Alice 108 Kilmarnock Bos 02215..........617 425-0193
Alice 45 Market Cambridge 02139..........617 945-2711
Andrew F 62 Vinal Av Som 02143..........617 625-7623
Carter Anne MD
1101 Beacon Bro 02446..........617 739-1022
Carter Athens
272 Newbury Boston 02116..........617 536-6329
B E 68 Gladeside Av Mat 02126..........617 296-6911
Carter Barbara L MD
Tufts-New England Medical Center Bos 02111
Call..........617 636-0051
Carter Becky Bos 02124..........617 523-4368
Bernard J
112 Gladstone E Bos 02128..........617 567-3430
Bithiah 25 Medway Dor 02124..........617 298-8713
Blake 26 Mt Vernon Bos 02108..........617 367-9931
Carter Broadcasting Co
20 Park Plz Bos 02116..........617 423-0210
Carter & Burgess Consultants Inc
23 East St Cam 02141..........617 225-0200
Carter C 2000 Commwlth Av Bri 02135..........617 782-2118
C 228 Faywood Av East Boston 02128..........617 569-1545
C 359 Harvard Cam 02138..........617 491-4822
C 610 Walk Hill Mat 02126..........617 296-6392
C & M 43 Burroughs Jam 02130..........617 524-9558

Carter F 24 Hillock Ros 02131..........617 327-1105
Faye & Ricky
357 Columbus Av Bos 02116..........617 437-7331
Francis S 134 Temple W Rox 02132..........617 323-6781
Franklin & Anne
221 Mt Auburn Cam 02138..........617 354-0798
Fred 42 Haverford Jam 02130..........617 524-3078
Fred 96 Hinckley Rd Mil 02186..........617 698-1343
G & R 8 Verdun Dor 02124..........617 436-8906
G T 27 Franklin Av Som 02145..........617 623-7121
Gayle 25 Frontenac Dor 02124..........617 825-0322
Geo S 115 Moss Hill Rd Jam 02130..........617 522-3215
George 125 Nashua Bos 02114..........617 367-9548
Carter Halliday Associate
107 S Street Bos 02111..........617 456-1689
Carter Harry F
26 Running Brk Rd W Rox 02132..........617 325-5465
Carter Hide Co Inc
146 Summer Bos 02110..........617 542-7987
Carter Hilary 61 Harvey Cam 02140..........617 876-2750
Horace
241 Walnut Av Roxbury 02119..........617 442-5307
Howard Jr 26 Notre Dme Rox 02119..........617 445-5552
J Cam..........617 354-2688
J 15 Chatham Bro 02446..........617 232-7990
J 518 Harvard Bro 02446..........617 730-9483
J 775 Vfw Pkwy West Roxbury 02132..........617 323-5574
Carter J Jacques MD
1 Brookline Pl Bro 02446..........617 735-8787
Carter J M
1410 Columbia Rd S Bos 02127..........617 464-1040
Carter J M Ornamental Ironworks
Call..........Pembroke TelNo 617 436-5353
Carter J Veal Co
48 Newmarket Sq Rox 02118..........617 442-1775
Carter James
1573 Cambridge St Cam 02138..........617 492-1214
James 182 Fisher Av Roxbury 02120..........617 739-2193
James
37 Gold Star Rd Cambridge 02140..........617 876-8841
Jas L 14 Roseberry Rd Mat 02126..........617 361-0773
Jane 114 Adena Rd Newton 02465..........617 964-0435
Jeffrey 41 Warren Av Bos 02116..........617 426-5994
John 11 Mansfield Bri 02134..........617 987-2163
John 327 Summer Bos 02210..........617 423-4334
John 40 Westwind Rd Dor 02125..........617 282-1235
June O 329 A Summit Av Bri 02135..........617 734-6109
K 38 Browning Av Dorchester 02124..........617 265-8456
K 17 Esmond Dorchester 02121..........617 282-1593

Carter Nella E
333 Maschsts Av Bos 02115..........617 267-6483
Nicholas S F
115 Randolph Av Mil 02186..........617 698-5307
Nick 21 Fairfield Bos 02116..........617 267-5222
Nick & Debbi
196 Herrick Rd Newton 02459..........617 527-0480
Nicole..........617 698-0713
Norman G
38 Chickatawbut Dor 02122..........617 822-1203
P 94 Crestwood Pk Rox 02121..........617 427-4754
P E 501 E Sixth S Bos 02127..........617 268-4213
P L 44 Hutchings Rox 02121..........617 427-9170
P R 91 Bynner Jam 02130..........617 983-8692
Paul & Constance
114 Anawan Av W Rox 02132..........617 325-2036
Paul E 501 E Sixth St S Bos 02127..........617 268-4546
Paul M 27 Union Bri 02135..........617 787-2115
Carter Pile Driving Inc 17 Beaver Ct
Framingham 01702..........Wellesley TelNo 781 235-8488
Carter Prudence
46 Franklin Watertown 02172..........617 393-3782
Prudence
46 Franklin Watertown 02172..........617 926-7063
Reginald
106 Brunswick Dorchester 02121..........617 541-2843
Renee & Andrew
10 Walnut Bos 02108..........617 720-3765
Carter Rice Dowd
Bulkley Dunton Publishing 163 Main Wilmington 01887
Toll Free-Dial "1" & Then..........800 638-1671
Cust Svc-Industrial Prod 613 Main Wilmington
Toll Free-Dial "1" & Then..........800 619-7447
Cust Svc-Printing 613 Main Wilmington
Toll Free-Dial "1" & Then..........800 648-7447
Headquarters 613 Main Wilmington 01887
Call..........978 988-7447
Ingalls Cronin 163 Main Wilmington 01887
Toll Free-Dial "1" & Then..........800 638-1673
Carter Richard
1079 Commwlth Av Brighton 02215..........617 987-0836
Richard A 97 Mt Vernon Bos 02108..........617 566-7293
Carter Richard A MD
170 Commwlth Av Bos 02116..........617 267-0710
Carter Richard K
15 Mercer S Bos 02127..........617 268-0448
Robert L 175 Richdale Av Cam 02140..........617 864-1535
Roger 150 St Botolph Bos 02115..........617 424-6148
Roy 44 Concord Av Cam 02138..........617 491-6115
Royce 18 Seminary Cha 02129..........617 241-0418

C

The Ithaca phone book has 10,000+ entries.

The Manhatten phone book has 1,000,000+ entries. But it does not take 100 x longer to look something up. Why?

Key Idea: Repeated Halving

To find Derek Jeter's number…

```
B = phone book
while (B is longer than 1 page):
    1. P = middle page of B
    2. Let Q be the first name on P
    3. if  'Jeter" comes before Q:
           Rip away the 2nd half of B
       else:
           Rip away the 1st half of B.

Scan remaining page P line-by-line for 'Jeter'
```

What Happens to Phone Book Length?

Original:	3000 pages
After 1 rip:	1500 pages
After 2 rips:	750 pages
After 3 rips:	375 pages
After 4 rips:	188 pages
After 5 rips:	94 pages
After 12 rips:	1 page

Binary Search

The idea of repeatedly halving the size of the "search space" is the main idea behind the method of binary search.

An item in a sorted array of length n can be located with approximately $\log_2 n$ comparisons.

$\log_2 8 = 3$ $\log_2 64 = 6$ $\log_2 2$**k = k

What is $\log_2(n)$?

n	ceil($\log_2(n)$)
10	4
100	7
1000	10
10000	14
100000	17
1000000	20

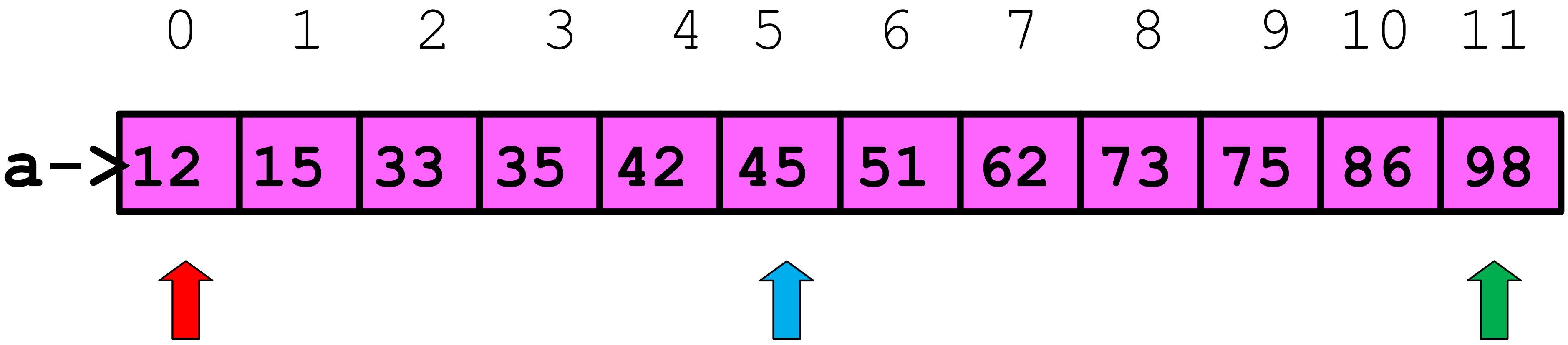
BinSearch: The Spec

```
def BinSearch(x,a):
    """ Returns an int k with the
    property that a[k]==x is True.
    If no such k exists, then
    k==-1.

    PreC: a is a nonempty list of
    ints that is sorted from smallest
    to largest. x is an int.
    """
```

Example:
Does this List have an Element With Value Equal to 70?

0	1	2	3	4	5	6	7	8	9	10	11
12	**15**	**33**	**35**	**42**	**45**	**51**	**62**	**73**	**75**	**86**	**98**

Let's Look For x in a



	0	1	2	3	4	5	6	7	8	9	10	11
a->	12	15	33	35	42	45	51	62	73	75	86	98

L: 0

Mid: 5

R: 11

x: 70

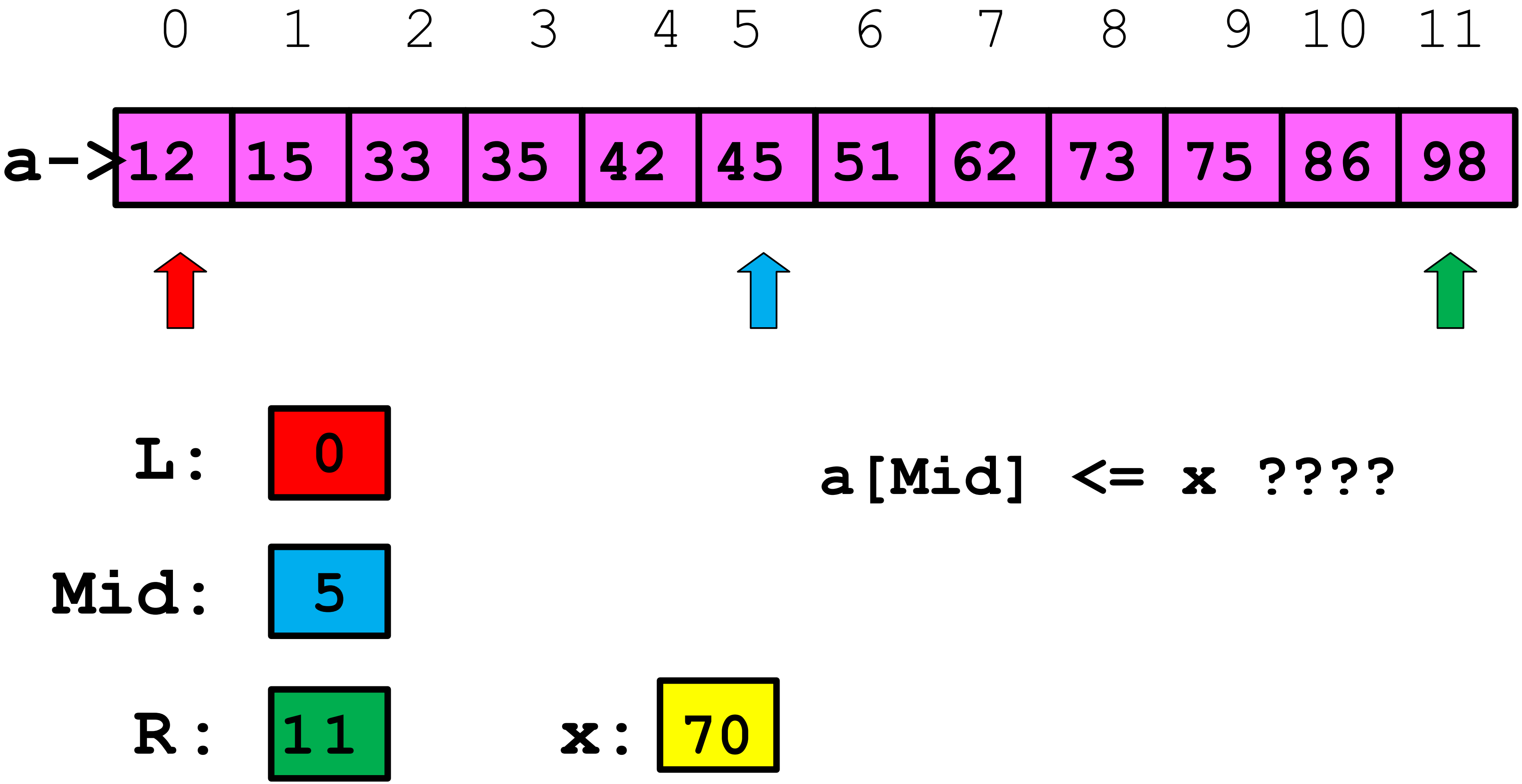
a[Mid] <= x ????

Mid = (L+R)/2

The Midpoint Computations

L	R	(L+R)/2
0	11	5
2	6	4
1	100	50

Let's Look For x in a



a->

0	1	2	3	4	5	6	7	8	9	10	11
12	15	33	35	42	45	51	62	73	75	86	98

L: 0

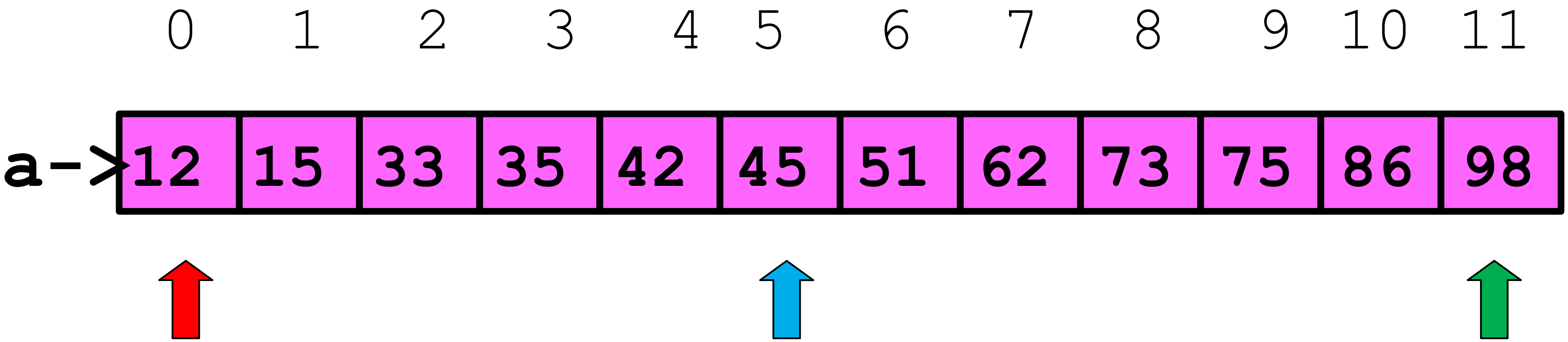
Mid: 5

R: 11

x: 70

a[Mid] <= x ????

Let's Look For x in a



	0	1	2	3	4	5	6	7	8	9	10	11
a->	12	15	33	35	42	45	51	62	73	75	86	98

`L:` 0

`Mid:` 5

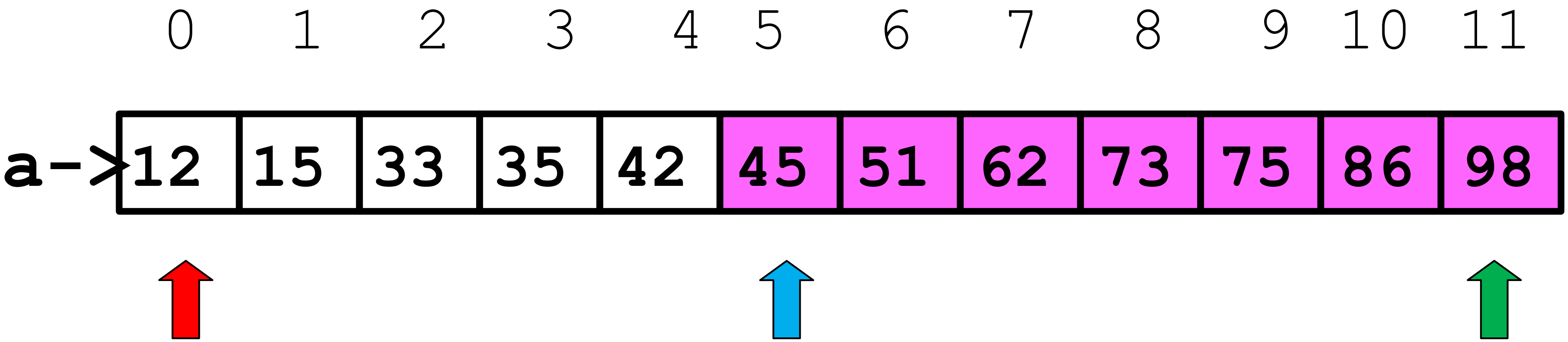
`R:` 11

`x:` 70

`a[Mid] <= x`

Yes!
So throw away
The "left half"

Let's Look For x in a



	0	1	2	3	4	5	6	7	8	9	10	11
a->	12	15	33	35	42	45	51	62	73	75	86	98

L: 0

Mid: 5

R: 11

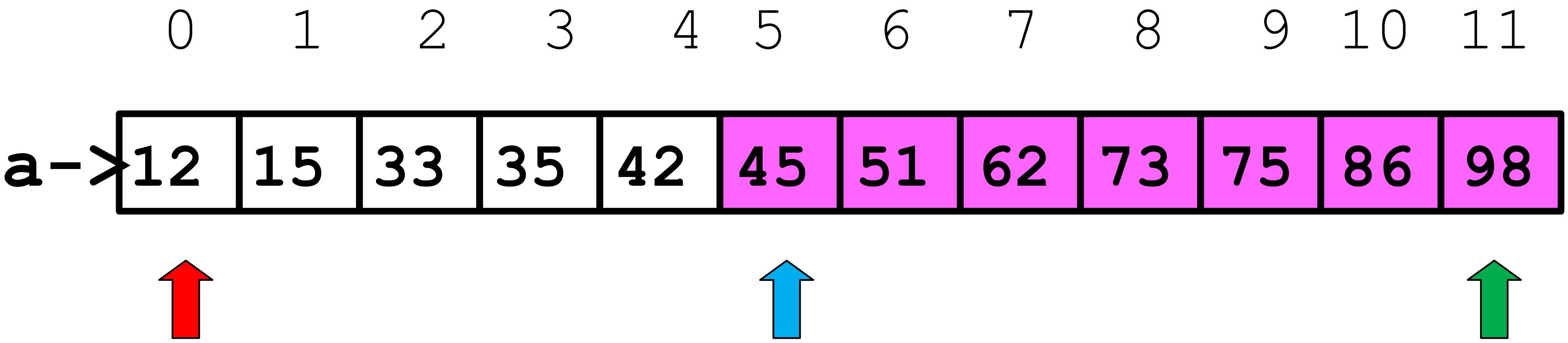
x: 70

`a[Mid] <= x`

Yes!

So throw away
The "left half"

Let's Look For x in a



	0	1	2	3	4	5	6	7	8	9	10	11
a->	12	15	33	35	42	45	51	62	73	75	86	98

L: 0

Mid: 5

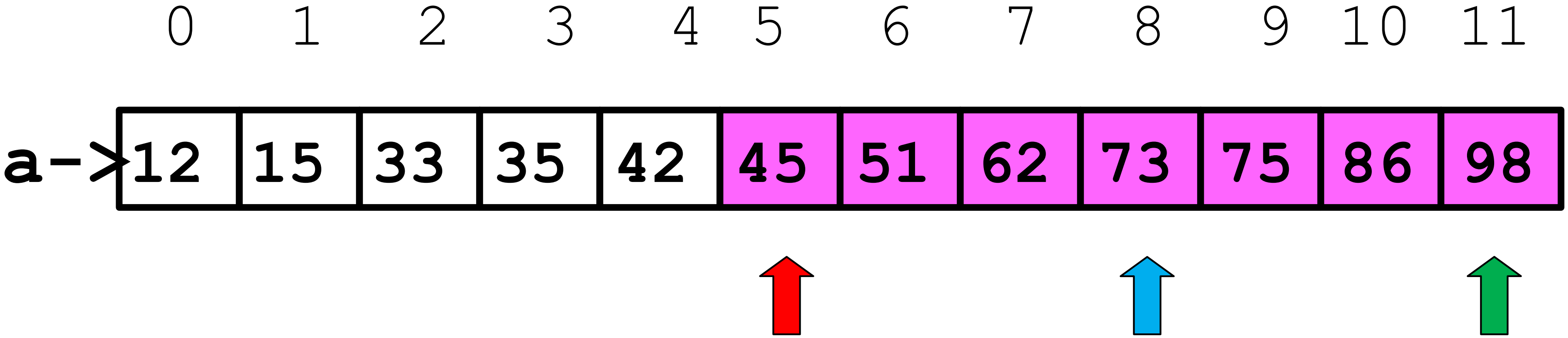
R: 11

x: 70

`a[Mid] <= x`

Revise L and Mid

Let's Look For x in a



	0	1	2	3	4	5	6	7	8	9	10	11
a->	12	15	33	35	42	45	51	62	73	75	86	98

L: 5

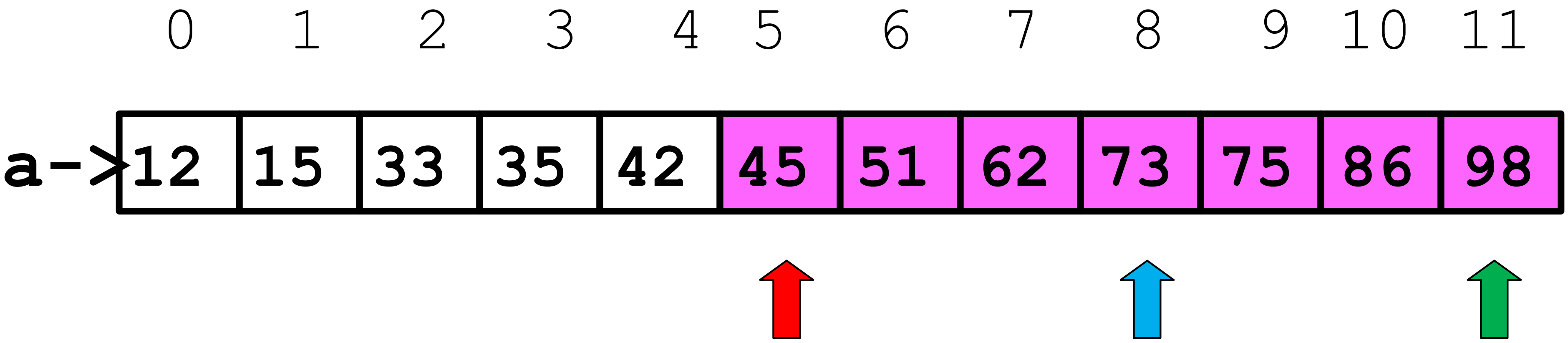
Mid: 8

R: 11

a[Mid] <= x ???

x: 70

Let's Look For x in a



	0	1	2	3	4	5	6	7	8	9	10	11
a->	12	15	33	35	42	45	51	62	73	75	86	98

L: 5

Mid: 8

R: 11

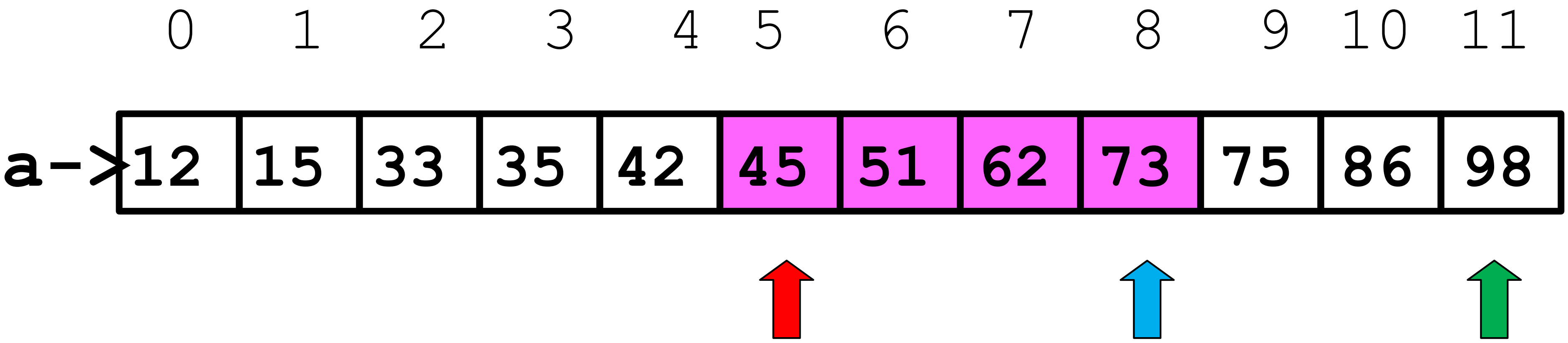
x: 70

`a[Mid] <= x`

No

So throw away the
“right half”

Let's Look For x = 70



	0	1	2	3	4	5	6	7	8	9	10	11
a->	12	15	33	35	42	45	51	62	73	75	86	98

L: 5

Mid: 8

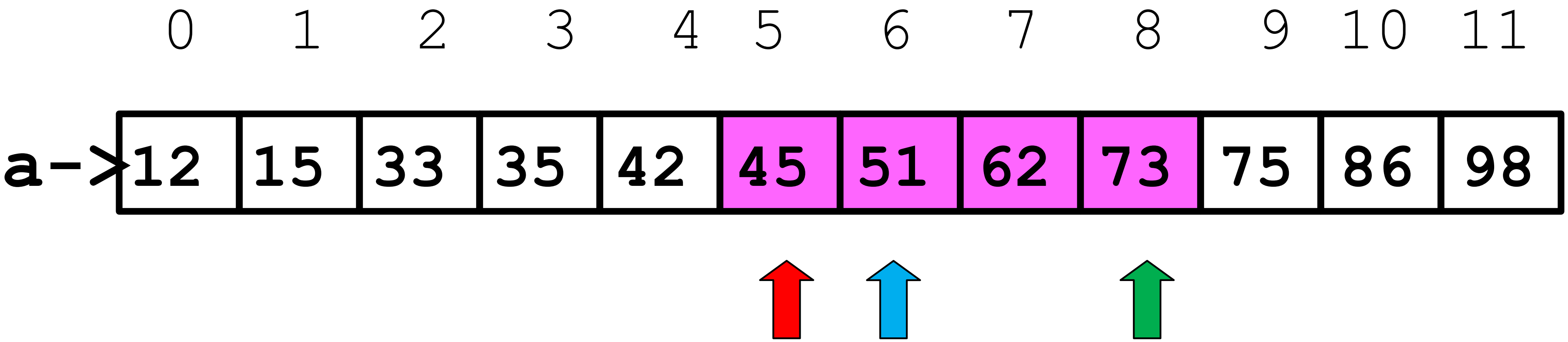
R: 11

x: 70

`a[Mid] <= x`

Revise R and Mid

Let's Look For x = 70



	0	1	2	3	4	5	6	7	8	9	10	11
a->	12	15	33	35	42	45	51	62	73	75	86	98

L: 5

Mid: 6

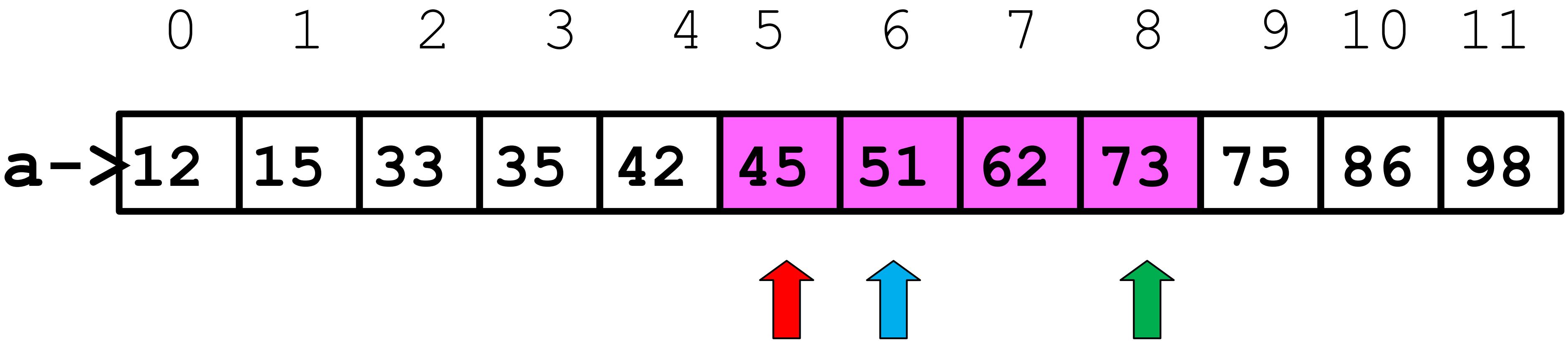
R: 8

x: 70

`a[Mid] <= x`

Revise R and Mid

Let's Look For x in a



	0	1	2	3	4	5	6	7	8	9	10	11
a->	12	15	33	35	42	45	51	62	73	75	86	98

L: 5

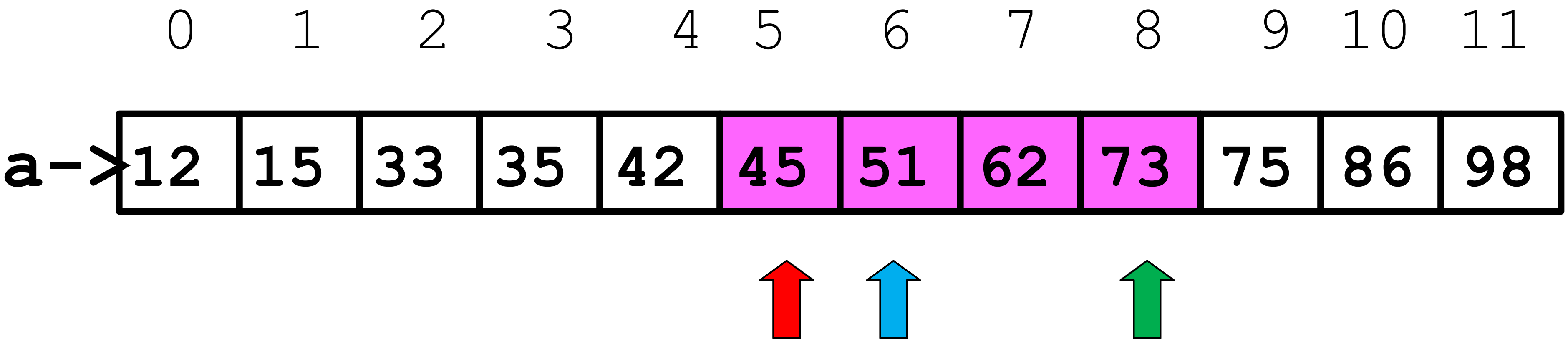
a[Mid] <= x ????

Mid: 6

R: 8

x: 70

Let's Look For x in a



	0	1	2	3	4	5	6	7	8	9	10	11
a->	12	15	33	35	42	45	51	62	73	75	86	98

L: 5

Mid: 6

R: 8

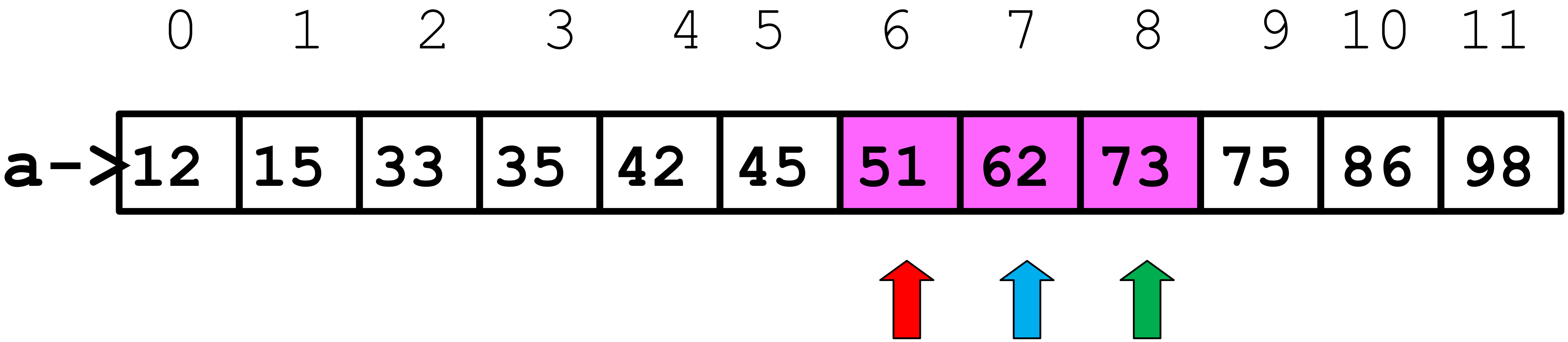
x: 70

`a[Mid] <= x`

Yes

Throw away the
Left half

Let's Look For x in a



	0	1	2	3	4	5	6	7	8	9	10	11
a->	12	15	33	35	42	45	51	62	73	75	86	98

L: 6

Mid: 7

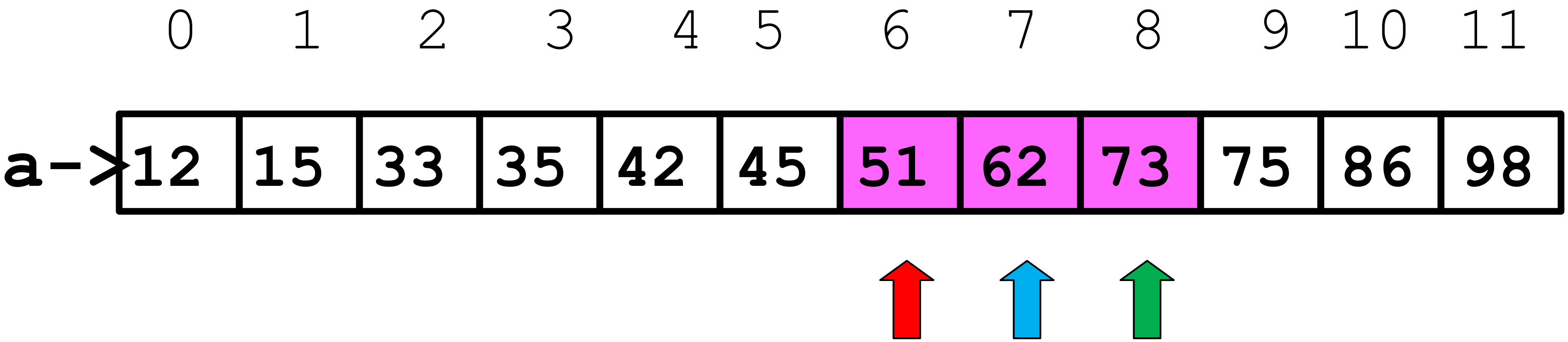
R: 8

x: 70

a[Mid] <= x

Yes

Let's Look For x in a



	0	1	2	3	4	5	6	7	8	9	10	11
a->	12	15	33	35	42	45	51	62	73	75	86	98

L: 6

Mid: 7

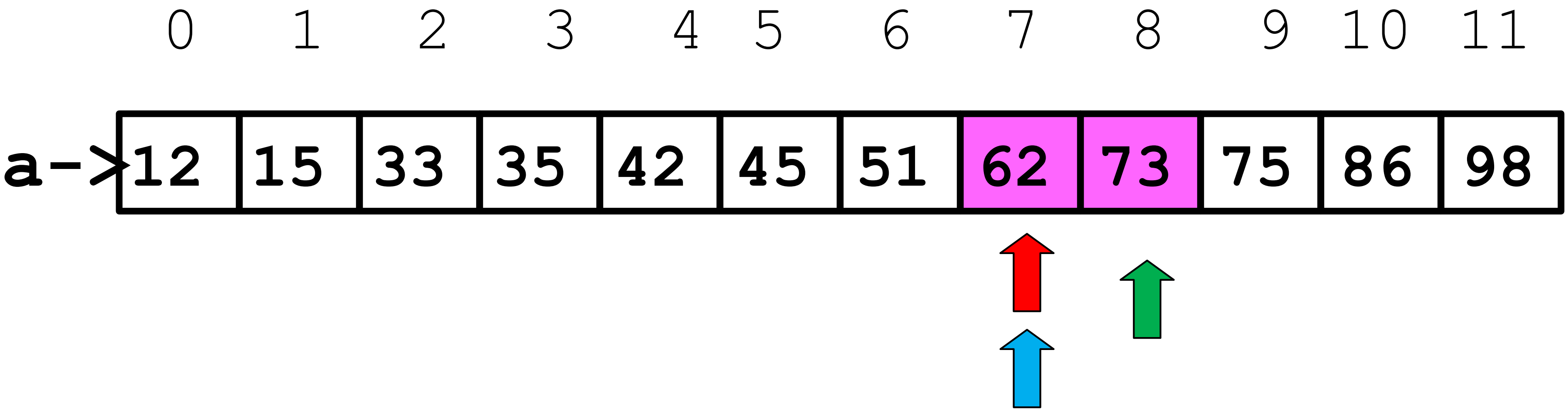
R: 8

x: 70

`a[Mid] <= x`

Throw away the left half

Let's Look For x in a



	0	1	2	3	4	5	6	7	8	9	10	11
a->	12	15	33	35	42	45	51	62	73	75	86	98

L: 6

Mid: 7

R: 8

x: 70

Done! At this point we just compare **x** with **a[L]** and **a[L+1]**.

What We Just Did

```
L = 0
R = len(a)-1
while R-L > 1:
  # a[L]<=x<=a[R]
  Mid = (L+R)/2
  if x <= a[mid]:
      R = Mid
  else:
      L = Mid
```

A Loop Invariant

Note that a[L]<=x<=a[R] remains True throughout the loop

What We Just Did

```
L = 0
R = len(a)-1
while R-L > 1:
    # a[L]<=x<=a[R]
    Mid = (L+R)/2
    if x <= a[mid]:
        R = Mid
    else:
        L = Mid
```

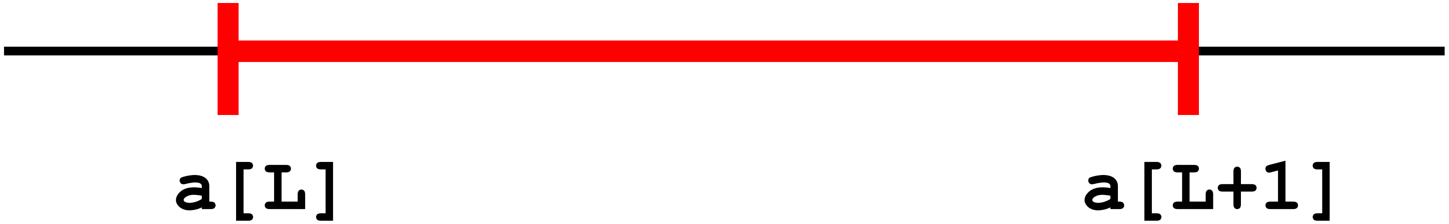
What is the situation when the loop terminates?

What We Just Did

```
L = 0
R = len(a)-1
while R-L > 1:
    # a[L]<=x<=a[R]
    Mid = (L+R)/2
    if x <= a[mid]:
        R = Mid
    else:
        L = Mid
```

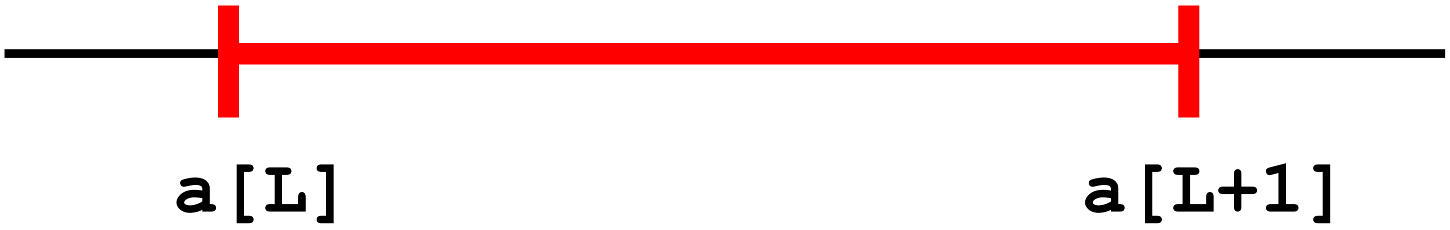
R-L<=1 implies R = L+1

After the Loop Ends



This is True: `a[L]<=x<=a[L+1]`

After the Loop Ends



```
if x==a[L]:
    return L
elif x==a[L+1]:
    return L+1
else:
    return -1
```

Measuring Execution Time

We now have two ways to search a list:

```
LinSearch(x,a)
BinSearch(x,a)
```

Intuition: BinSearch much faster.

Can we quantify this with a “stop watch”?

The timeit Module

This module can be used to time how long it takes to execute a chunk of code.

Typical chunk = some function of interest.

This is called benchmarking.

Benchmarking

Let's benchmark **`LinSearch(x,a)`** and **`BinSearch(x,a)`**.

Compare how long it takes when **`len(a)`** equals 1000, 10000, 100000, and 1000000.

Our intuition tells us that as **`len(a)`** increases, **`BinSearch`** will be dramatically faster.

BinSearch vs LinSearch

n	tBin	tLin	tLinW
1000	0.0007	0.0064	0.0119
10000	0.0009	0.0668	0.1203
100000	0.0011	0.8296	1.2082
1000000	0.0015	17.7388	13.9341

```
tBin  = time for BinSearch
tLin  = time for LinSearch (for loop version)
tLinW = time for LinSearch (while-loop version)
```

BinSearch vs LinSearch

n	tLin/tBin
1000	9
10000	74
100000	754
1000000	7095

Reporting ratios is more illuminating since we do not really care about the time units in this informal comparison

Using the `timeit` Module

We show how this module was used to get the results on the previous slides.

Our LinSearch vs BinSearch example is very typical: is one function faster than another?

A Benchmarking Framework

```
from timeit import *
S = “““
   Set-up code
”””
B = “““
   Code to Benchmark
”””
p = 10; m = 100
t = min(Timer(B,setup=S).repeat(p,m))
```

Yes, these are doc strings.

The Set-Up and Bench Codes

```
from random import randint as randi
from ShowSearch import BinSearch
n = 10000
s = [randi(0,10*n) for i in range(n)]
s.sort()
x = s[n/2]
```

```
k=BinSearch(x,s)
```

The set-up code is run once.

It is not timed.

It just sets up the code to be timed.

A Benchmarking Framework

```
from timeit import *
S = “““
    Set-up code
”””
B = “““
    Code to Benchmark
”””
p = 10; m = 100
t = min(Timer(B,setup=S).repeat(p,m))
```

An “experiment” consists of running the blue code m times.

The stopwatch will time how long it takes to do one experiment

Larger values necessary if the blue code executes very quickly

A Benchmarking Framework

```
from timeit import *
S = “““
    Set-up code
”””
B = “““
    Code to Benchmark
”””
p = 10; m = 100
t = min(Timer(B,setup=S).repeat(p,m))
```

Timer returns a length-p list. Each element is the stopwatch time for 1 experiment

This helps control for other stuff that may be running on your computer.

A Benchmarking Framework

```
from timeit import *
S = """
    Set-up code
”””
B = """
    Code to Benchmark
”””
p = 10; m = 100
t = min(Timer(B,setup=S).repeat(p,m))
```

In general, it is best to take the mininum as the most reliable. The benchmark time is assigned to t

This helps control for other stuff that may be running on your computer.

Why Benchmarking is Important

Confirms/refutes what our intuition might say about efficiency.

Makes us sensitive to the various issues that affect efficiency.

Steers us away from simplistic comparisons of different methods that can be used on the same problem.

Demonstration!

- Download and run the python script ShowBenchmarking.py
- TODO:
  - Run the above script for the following values of “n” (line 55)
    - 1000, 10000, 100000, 500000, 1000000, 10000000
  - Of the three search methods—binary search, linear search with for loop, linear search with while—which seems the fastest?
  - Which is faster of the following two—linear search with for loop, linear search with while loop?