MCAC 301

Design and Analysis of Algorithms

Neelima Gupta

ngupta@cs.du.ac.in

PseudoCode
Eg. Linear Search

HOD's Office

Makes request for file labeled "MCA admissions 2019"

Secretary's office containing Unorganised Files

Secretary responds with "File Not Found"

Figure A: Secretary at an office responding to the requests of the Head

- If there were n files and the requested file was not found, how many comparisons did the secretary have to perform?

Definition 1. ***Searching***: *Given a set of elements* $\{x_1, x_2 .... x_n\}$ *stored in an array and a target element* x*, problem is to find the location/index where* x *is found in the array or return* 0 *otherwise.*

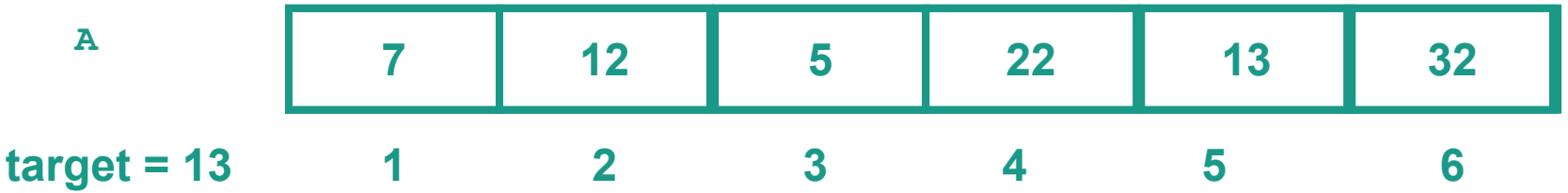
Consider the following algorithm that examines each element sequentially to determine if it is x or not. It is called **Linear/Sequential Search**.

Algorithm 1: Linear search

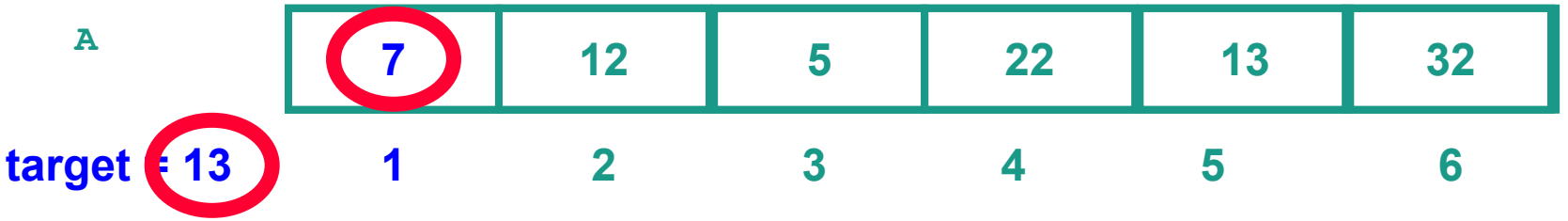
```
   input  : Array: A[1...n], Key
   output: index of first occurrence of key if it is found, 0 otherwise
1  for i ← 1 to n do
2  |   if A[i] = key then
3  |   |   return i
4  |   end
5  end
6  return 0
7
```

Note: We use indices starting from 1 to n in algorithms (pseudocode) and not 0 to n-1

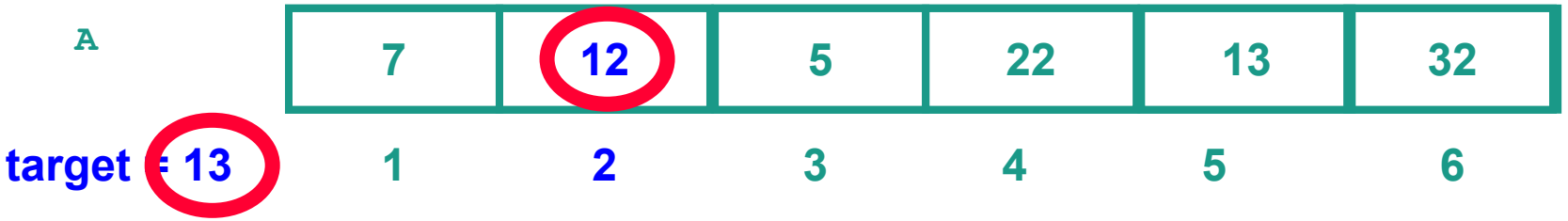
Example of Linear Search



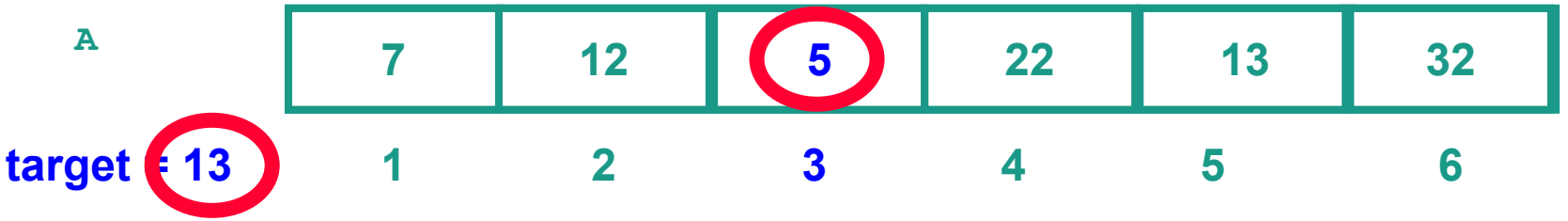
https://www.cc.gatech.edu/home/bleahy/cs1311/cs1311lecture15wdl.



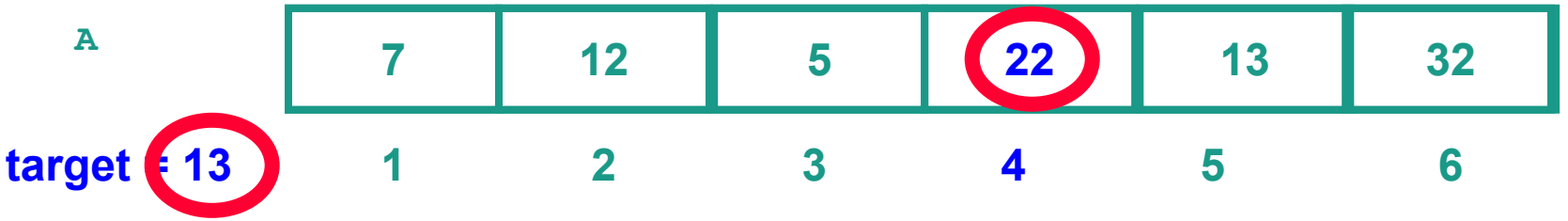
https://www.cc.gatech.edu/home/bleahy/cs1311/cs1311lecture15wdl.



https://www.cc.gatech.edu/home/bleahy/cs1311/cs1311lecture15wdl.

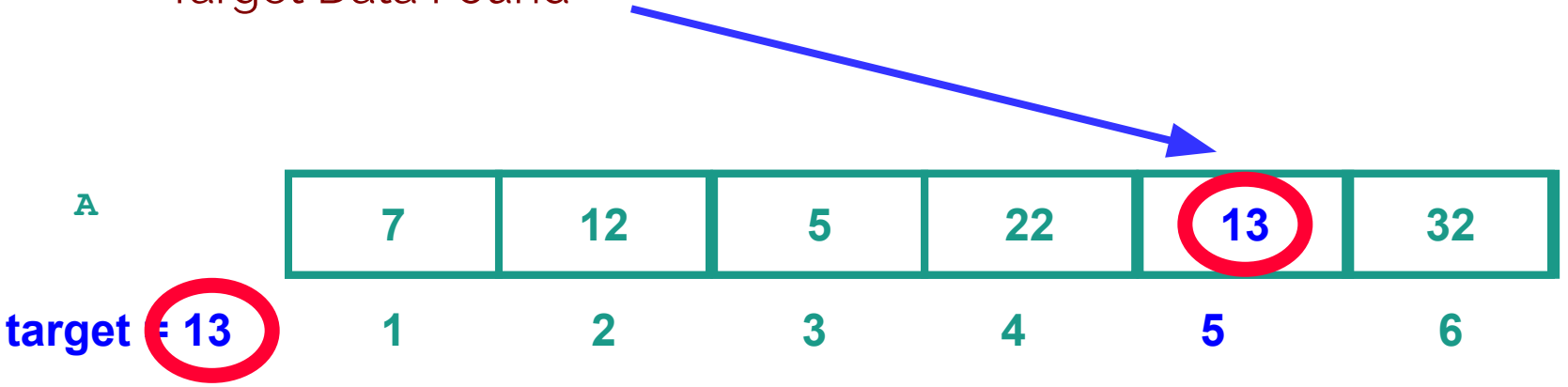


https://www.cc.gatech.edu/home/bleahy/cs1311/cs1311lecture15wdl.



https://www.cc.gatech.edu/home/bleahy/cs1311/cs1311lecture15wdl.

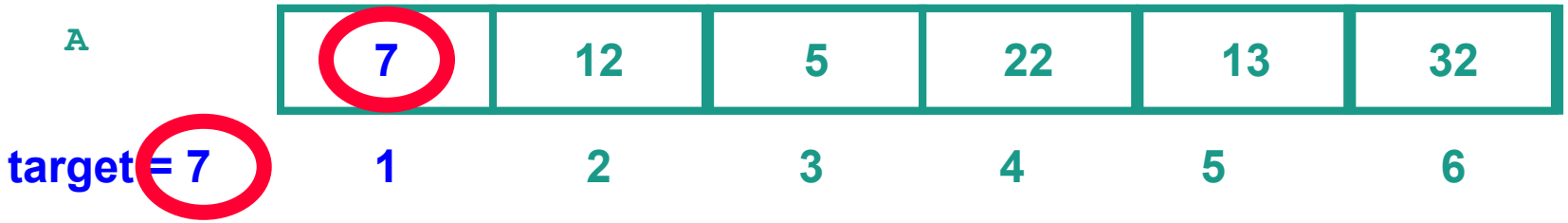
Target Data Found



https://www.cc.gatech.edu/home/bleahy/cs1311/cs1311lecture15wdl.

Linear Search Analysis: Best Case

Best Case:
1 comparison

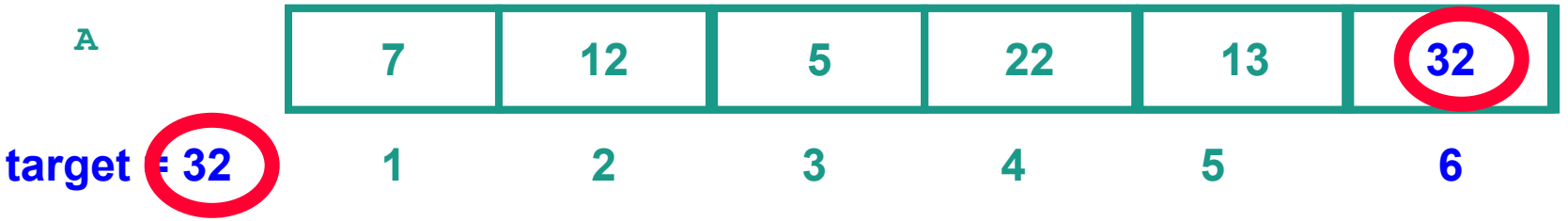


Best Case: match with the first item

https://www.cc.gatech.edu/home/bleahy/cs1311/cs1311lecture15wdl.

Linear Search Analysis: Worst Case

Worst Case:
N comparisons

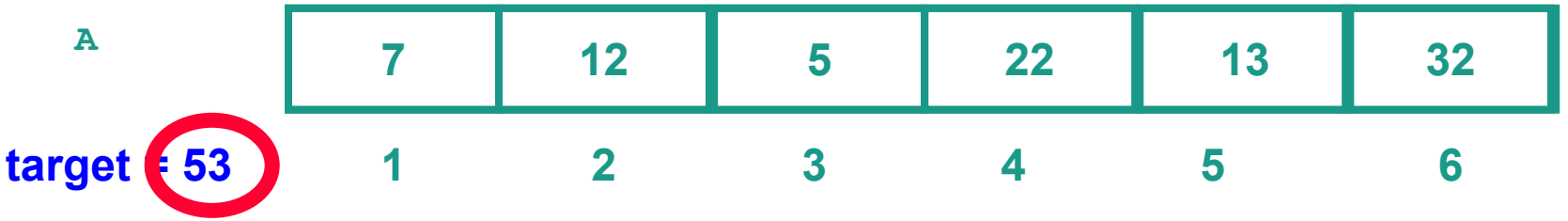


Worst Case: match with the last item

https://www.cc.gatech.edu/home/bleahy/cs1311/cs1311lecture15wdl.

Linear Search Analysis: Unsuccessful Search Case

Unsuccessful Search: N comparisons



Unsuccessful Search: no match

https://www.cc.gatech.edu/home/bleahy/cs1311/cs1311lecture15wdl.

Analysis of Linear Search : Counting the number of primitive steps

Algorithm 1: Linear search

```
   input  : Array: A[1...n], Key
   output: index of first occurrence of key if it is found, 0 otherwise
1  for i ← 1 to n do
2  |   if A[i] = key then
3  |   |   return i
4  |   end
5  end
6  return 0
7
```

T1

T2

1. What is the relationship between T1 and T2?
   Ans: T1 = T2 for a successful search, T1 = T2 + 1 = n + 1 for an unsuccessful search
2. In the worst case successful search: T1 is ? T2 is?
   Ans: T1 = T2 =n
3. In the best case successful search: T1 is ? T2 is?
   Ans: T1 = T2 =1
4. In case of an unsuccessful search: T1 is ? T2 is?
   Ans: T1 = T2 + 1 = n + 1

Analysis of Linear Search : Counting the number of primitive steps

Total Number of Primitive Steps: (counting “assignment and comparison” done in statement 1 as one primitive step)

- T1+ T2 + 1(for the “return” statement)

this is =

- n + n + 1 = 2n + 1 in the worst case successful search
- 1 + 1 + 1 = 3 in the best case successful search
- (n + 1) + n + 1 = 2n + 2 in an unsuccessful search

Ignoring the constants

When we compare running times of two algorithms, constants become insignificant for large n.

For example

$\lim_{n \to \infty} \frac{(c1 \log n + c2)}{c3\, n + c4} = 0$, irrespective of the constants c1, c2, c3 and c4.

We will define the notion of asymptotic functions to capture this.

Programming Assignment:

- Implement Linear Search
  - Count the number of key comparisons for various inputs and plot the graph.
  - For every input size n , run it with n+ 1 different keys - n successful plus 1 unsuccessful
  - Compute the minimum, maximum and average of the number of key comparisons for each input size.
  - Plot the graph for each case - best, worst and average number of comparisons.
  - n varies from 10 to 100 in steps of 5