THE UNIVERSITY OF MELBOURNE

COMP90038
Algorithms and Complexity

Assignment Project Exam Help

Lectur https://eduassistpro.github.io/ ethods

(with thanks to Hara Add WeChat edu_assist_pro ergaard)

Toby Murray

toby.murray@unimelb.edu.au

DMD 8.17 (Level 8, Doug McDonell Bldg)

http://people.eng.unimelb.edu.au/tobym

@tobycmurray

Compulsory Quizzes

THE UNIVERSITY OF MELBOURNE

- **Remember:** you need to **complete** 8 of the quizzes to pass the hurdle

- By "completing" a quiz, we mean **getting all answers right (i** Assignment Project Exam Help **ting**

https://eduassistpro.github.io/

- You can have as many att Add WeChat edu_assist_pro s you need, but on at least one of those attempts you need to score 100%

- The first compulsory quiz (for week 2) closes **tomorrow**

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Brute Force Algorithms

THE UNIVERSITY OF MELBOURNE

- Straightforward problem solving approach, usually based directly on the problem's statement.

- Exhaustive search for solutions is a prime example.
  - Selection sort
  - String matching
  - Closest pair
  - Exhaustive search for combinatorial solutions
  - Graph traversal

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Example: Selection Sort

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

// sw and A[min]

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Example: Selection Sort

THE UNIVERSITY OF MELBOURNE

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

// sw and A[min]

Time Complexity:

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Example: Selection Sort

THE UNIVERSITY OF MELBOURNE

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

// sw and A[min]

Time Complexity: $\Theta(n^2)$

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Example: Selection Sort

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

// sw and A[min]

Time Complexity: $\Theta(n^2)$

We will soon meet better sorting algorithms

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Properties of Sorting Algorithms

- A Sorting algorithm is:
  - **in-place** if it does not require additional memory except, perhap s of memory
  - **stable** if it preserves the order of elements with identical keys
  - **input-insensitive** if its running time is fairly independent of input properties other than size

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Properties of Selection Sort

- While running time is quadratic, selection sort makes only about ***n*** **exchanges**.

- So: **selection sort is a good algorithm for sorting small collection** [illegible] **ds.**

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

- In-place?

- Stable?

- Input-insensitive?

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Properties of Selection Sort

THE UNIVERSITY OF MELBOURNE

- While running time is quadratic, selection sort makes only about ***n*** **exchanges**.
- So: **selection sort is a good algorithm for sorting small collection ds.**

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

- In-place? yes
- Stable?
- Input-insensitive?

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Properties of Selection Sort

THE UNIVERSITY OF MELBOURNE

- While running time is quadratic, selection sort makes only about ***n*** **exchanges**.

- So: **selection sort is a good algorithm for sorting small collection ds.**

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

- In-place? yes

- Stable? ?

- Input-insensitive?

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Properties of Selection Sort

THE UNIVERSITY OF MELBOURNE

- While running time is quadratic, selection sort makes only about ***n*** **exchanges**.

- So: **selection sort is a good algorithm for sorting small collection ds.**

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

- In-place? yes

- Stable? ?

- Input-insensitive? yes

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Selection Sort Stability

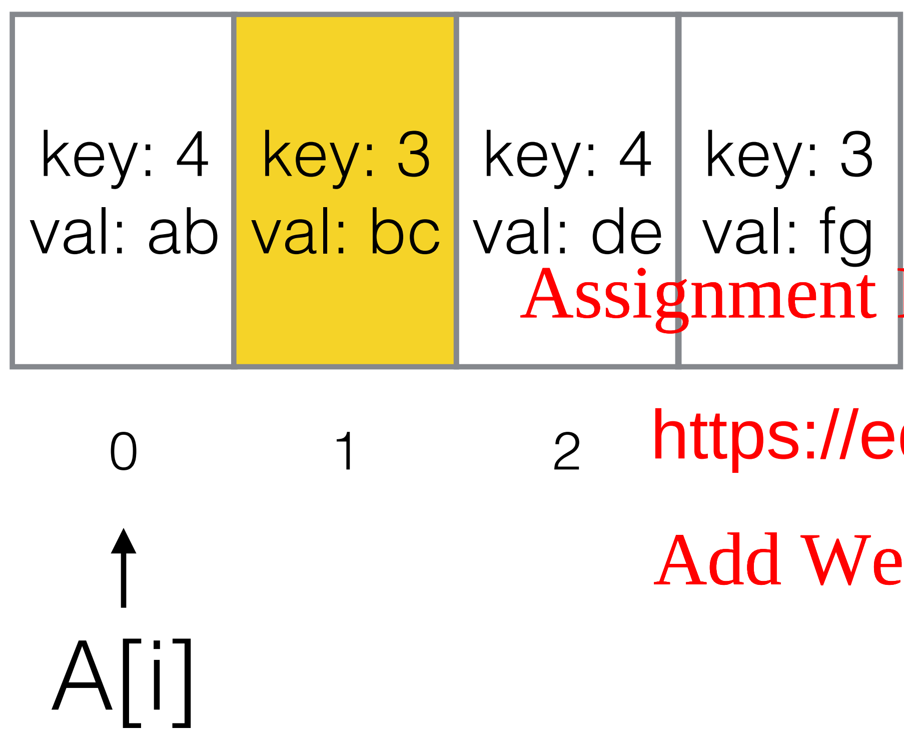
key: 4 val: ab	key: 3 val: bc	key: 4 val: de	key: 3 val: fg
0	1	2	

↑
A[i]

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Selection Sort Stability



Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Selection Sort Stability

key: 3 val: bc	key: 4 val: ab	key: 4 val: de	key: 3 val: fg
0	1	2	

↑
A[i]

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Selection Sort Stability

key: 3 val: bc	key: 4 val: ab	key: 4 val: de	key: 3 val: fg
0	1	2	

A[i]

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Selection Sort Stability

0	1	2	3
key: 3 val: bc	key: 4 val: ab	key: 4 val: de	key: 3 val: fg

↑
A[i]

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Selection Sort Stability

key: 3 val: bc	key: 3 val: fg	key: 4 val: de	key: 4 val: ab
0	1	2	

↑

A[i]

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Selection Sort Stability

key: 3 val: bc	key: 3 val: fg	key: 4 val: de	key: 4 val: ab

0 1 2

↑
A[i]

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

the relative order
of the two "4" records
has changed!

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Properties of Selection Sort

- While running time is quadratic, selection sort makes only about ***n*** **exchanges**.

- So: **selection sort is a good algorithm for sorting small collection ds.**

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

- In-place? yes

- Stable? no

- Input-insensitive? yes

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Brute Force String Matching

- **Pattern** *p*: a string of *m* characters to search **for**
- **Text** *t*: a long string of *n* characters to search **in**
- We use *i* to run through the text and *j* to run through the pattern

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Brute Force String Matching

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

t:

N	O	B	O	D	Y		K	N	O	W	S	
0	1	2	3	4	5	6	7	8	9	10	11	m

p:

N	O	T	
0	1	2	n

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Brute Force String Matching

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

t[i]

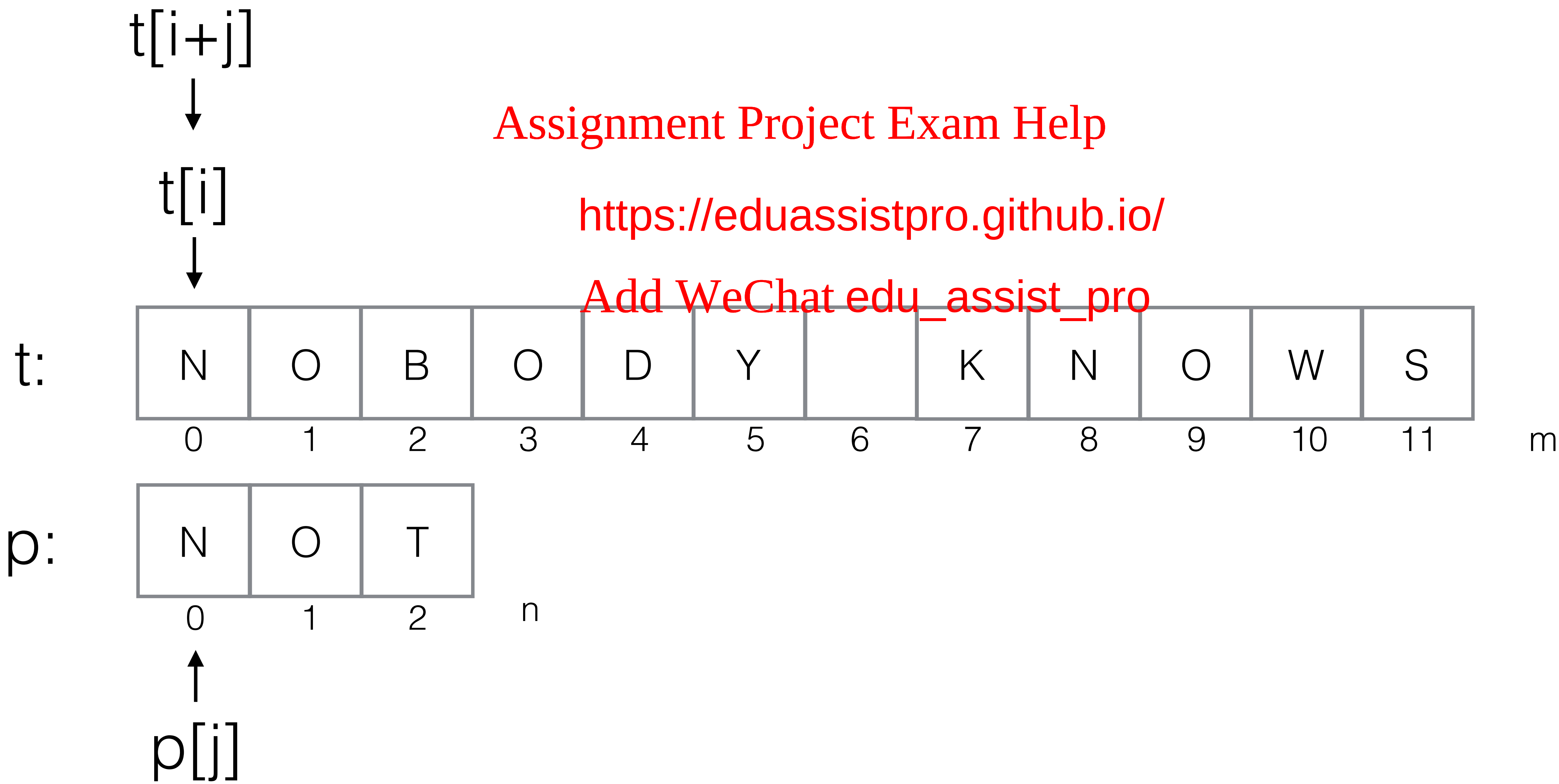
↓

	0	1	2	3	4	5	6	7	8	9	10	11	m
t:	N	O	B	O	D	Y		K	N	O	W	S	

	0	1	2	n
p:	N	O	T	

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Brute Force String Matching



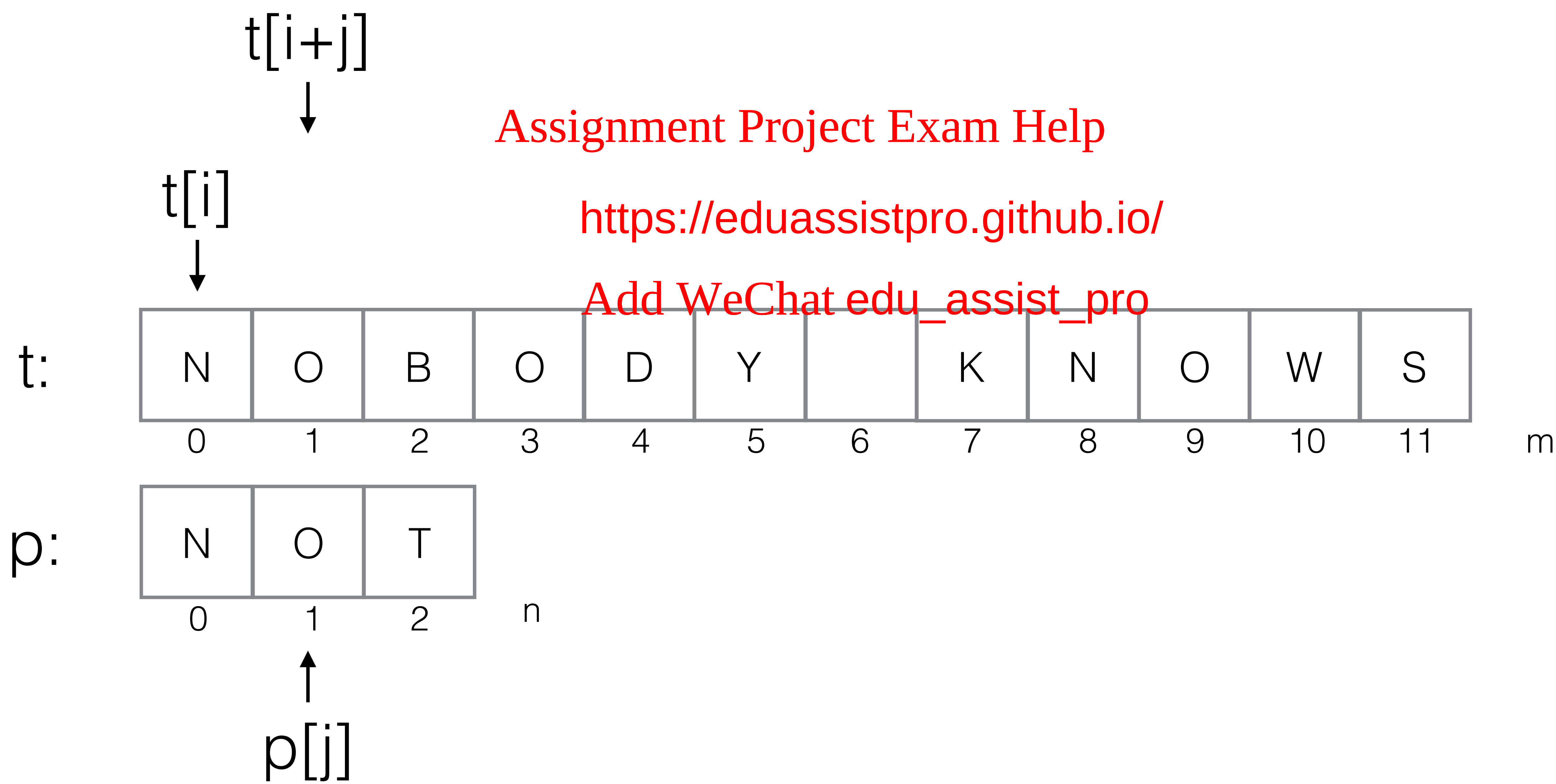
Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

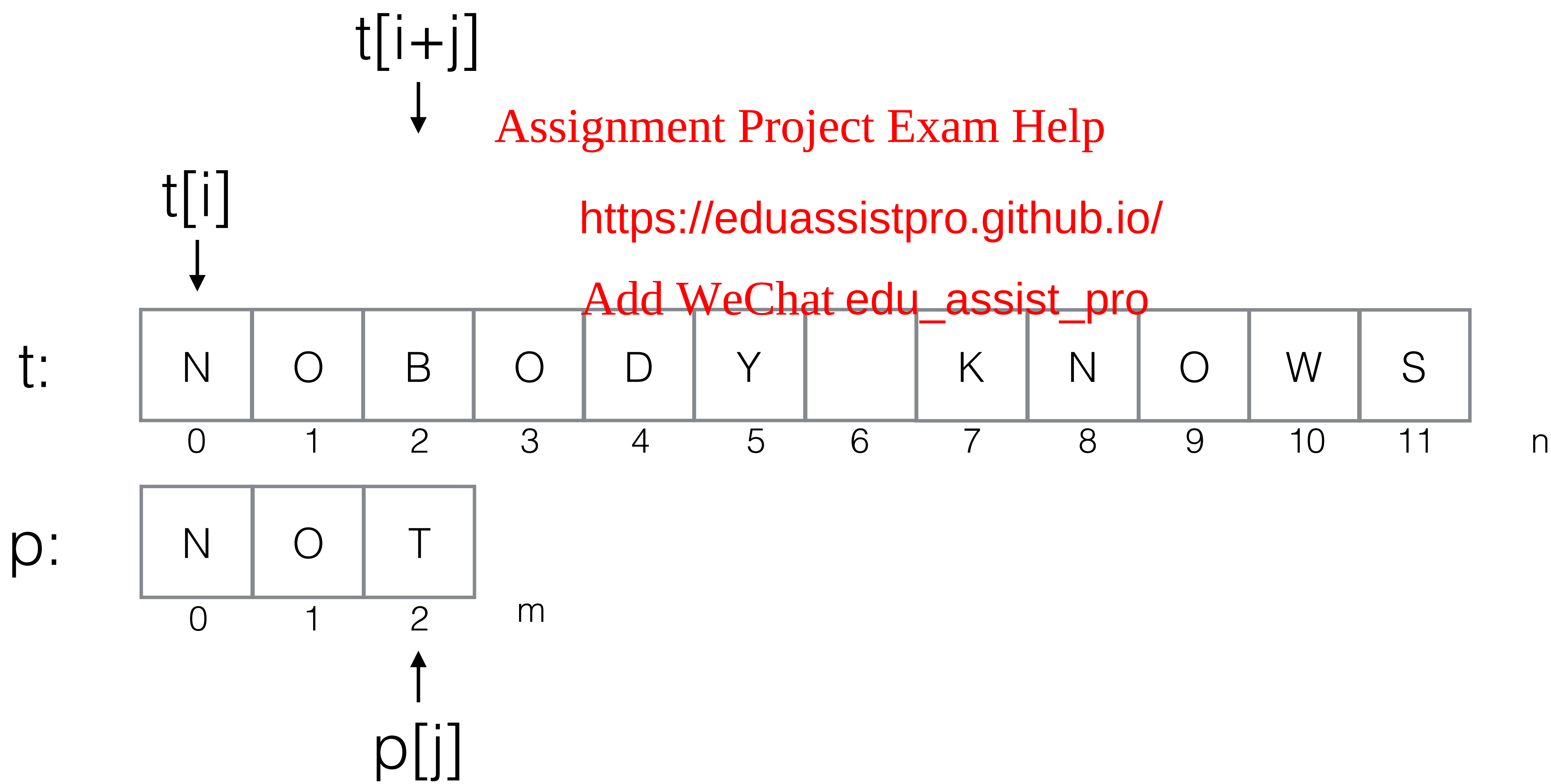
Brute Force String Matching



Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Brute Force String Matching



Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Brute Force String Matching

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

t[i]

↓

t:	N	O	B	O	D	Y		K	N	O	W	S	
	0	1	2	3	4	5	6	7	8	9	10	11	n

p:	N	O	T	
	0	1	2	m

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Brute Force String Matching

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

t[i]

t:	N	O	B	O	D	Y		K	N	O	W	S	
	0	1	2	3	4	5	6	7	8	9	10	11	n

p:	N	O	T	
	0	1	2	m

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Brute Force String Matching

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

t[i]

t:

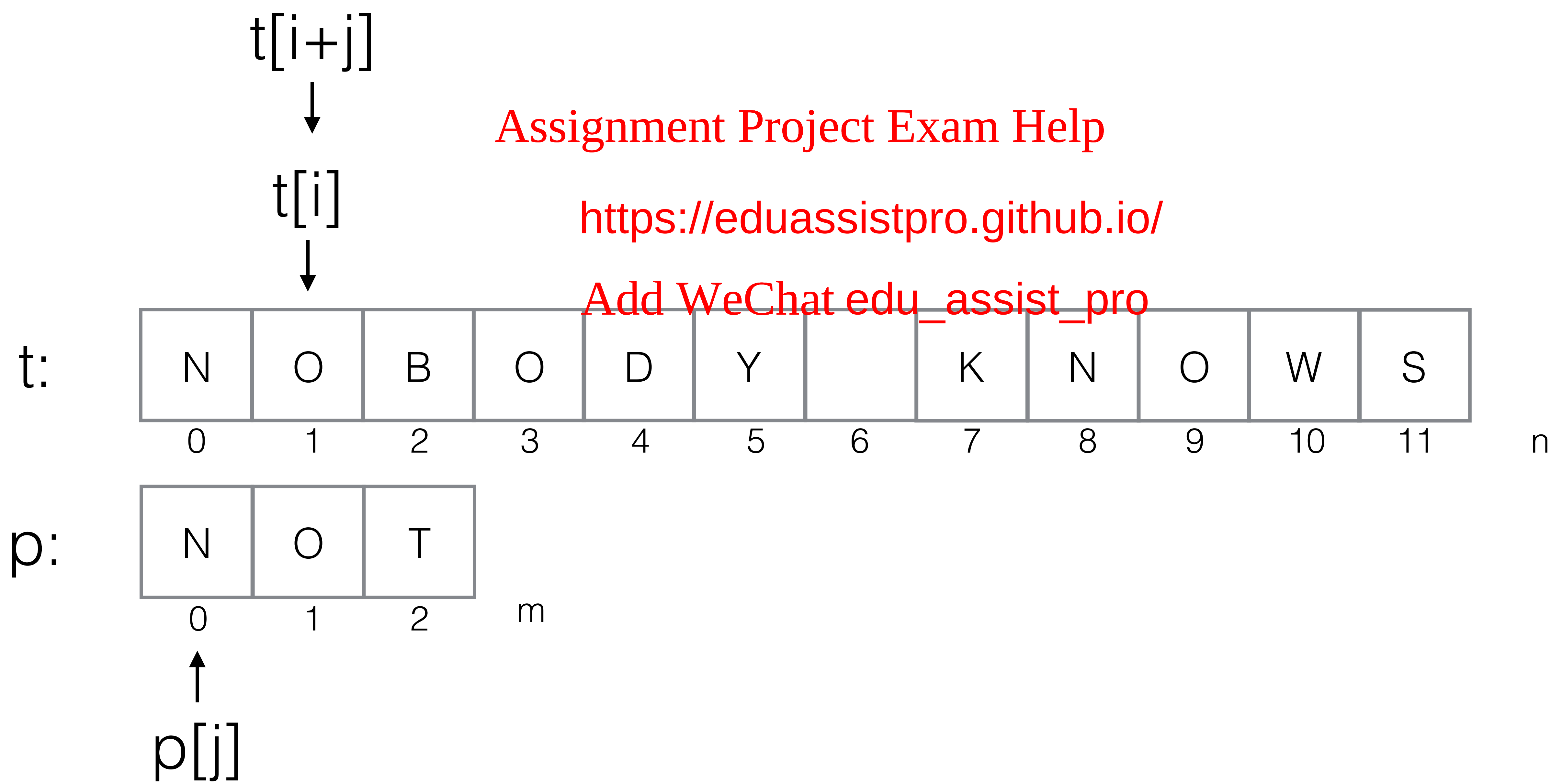
N	O	B	O	D	Y		K	N	O	W	S	
0	1	2	3	4	5	6	7	8	9	10	11	n

p:

N	O	T	
0	1	2	m

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Brute Force String Matching



Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

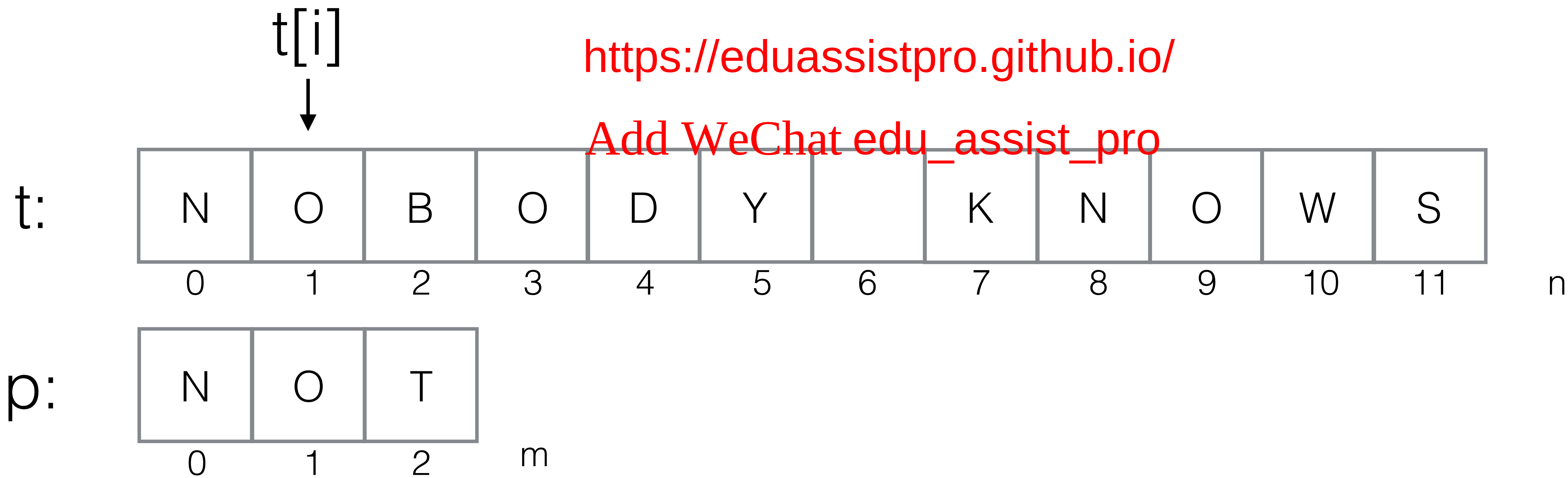
Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Brute Force String Matching

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro



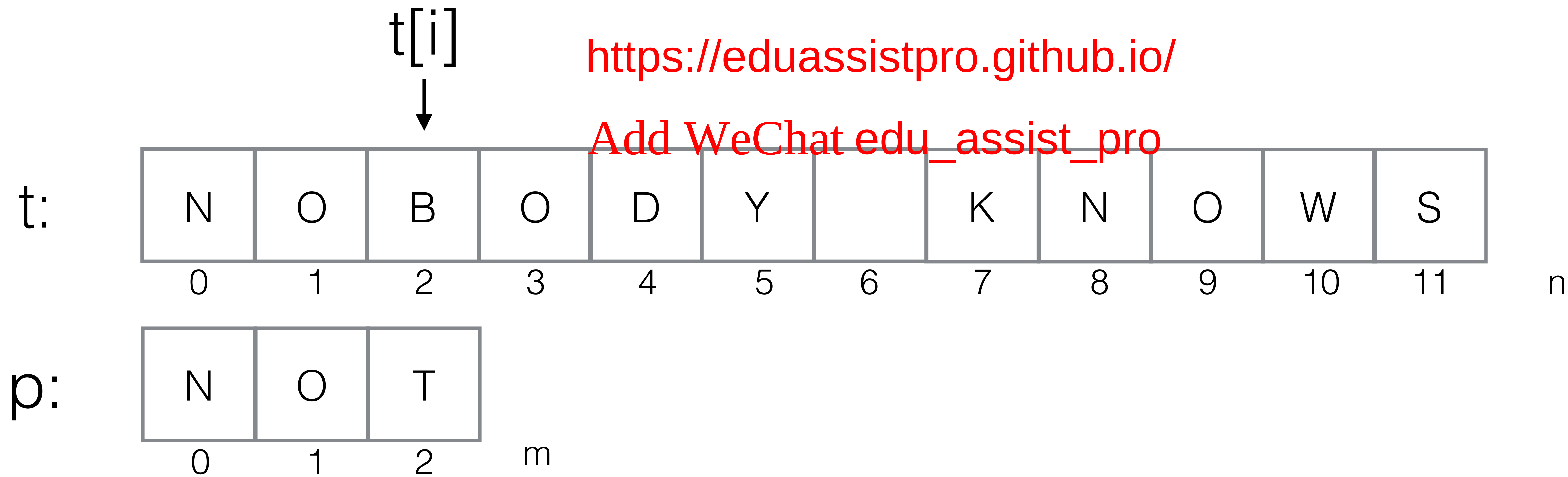
 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Brute Force String Matching

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro



Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Analysing Brute Force String Matching

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Analysing Brute Force String Matching

Basic operation:

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Analysing Brute Force String Matching

Basic operation: comparison p[j] = t[i+j]

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Analysing Brute Force String Matching

Basic operation: comparison p[j] = t[i+j]

For each n - m + 1 positions in *t*
we make up to m comparisons

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Analysing Brute Force String Matching

THE UNIVERSITY OF MELBOURNE

Basic operation: comparison p[j] = t[i+j]

For each n - m + 1 positions in *t*
we make up to m comparisons

Assignment Project Exam Help

Assuming m much https://eduassistpro.github.io/ (mn) comparisons.

Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Analysing Brute Force String Matching

THE UNIVERSITY OF MELBOURNE

Basic operation: comparison p[j] = t[i+j]

For each n - m + 1 positions in *t*
we make up to m comparisons

Assignment Project Exam Help

Assuming m much https://eduassistpro.github.io/ (*mn*) comparisons.

Add WeChat edu_assist_pro

But for **random text** over r ly large alphabet
(e.g. English), **average** running time is linear in *n*

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Analysing Brute Force String Matching

Basic operation: comparison p[j] = t[i+j]

For each n - m + 1 positions in *t*
we make up to m comparisons
Assignment Project Exam Help

Assuming m much https://eduassistpro.github.io/ (mn) comparisons.

Add WeChat edu_assist_pro
But for **random text** over r ly large alphabet
(e.g. English), **average** running time is linear in *n*

There are better algorithms, for smaller alphabets such as binary strings or strings of DNA nucleobases. But for many purposes, the brute-force algorithm is acceptable.

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Brute Force Geometric Algorithms: Closest Pair

THE UNIVERSITY OF MELBOURNE

- **Problem:** Given n points is k-dimensional space, find a pair of points with minimal separating Euclidean distance.

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

- The brute force a ers each pair in turn (except that d the distance from x to y, it does not nee sider the distance from y to x).

- For simplicity, we look at the 2-dimensional case, the points being $(x_0, y_0), (x_1, y_1), \ldots, (x_{n-1}, y_{n-1})$.

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Closest Pair Problem (2D)

THE UNIVERSITY OF MELBOURNE

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Closest Pair Problem (2D)

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Brute Force Closest Pair

Try all combinations (x_i, y_i) and (x_j, y_j) with $i < j$:

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Closest Pair Problem (2D)

2

Assignment Project Exam Help

0

https://eduassistpro.github.io/

3

Add WeChat edu_assist_pro

1

4

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Closest Pair Problem (2D)

i=0

2

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

0

3

1

4

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Closest Pair Problem (2D)

i=0 min=∞

2

Assignment Project Exam Help

0

https://eduassistpro.github.io/

3 Add WeChat edu_assist_pro

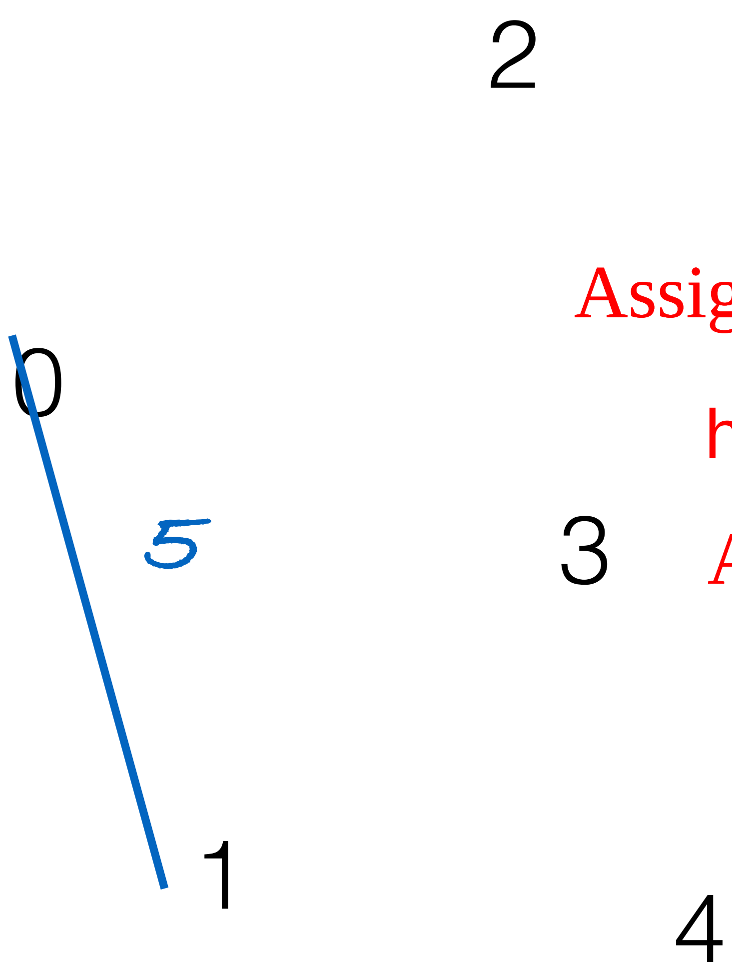
1

4

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Closest Pair Problem (2D)

i=0 min=∞



Assignment Project Exam Help

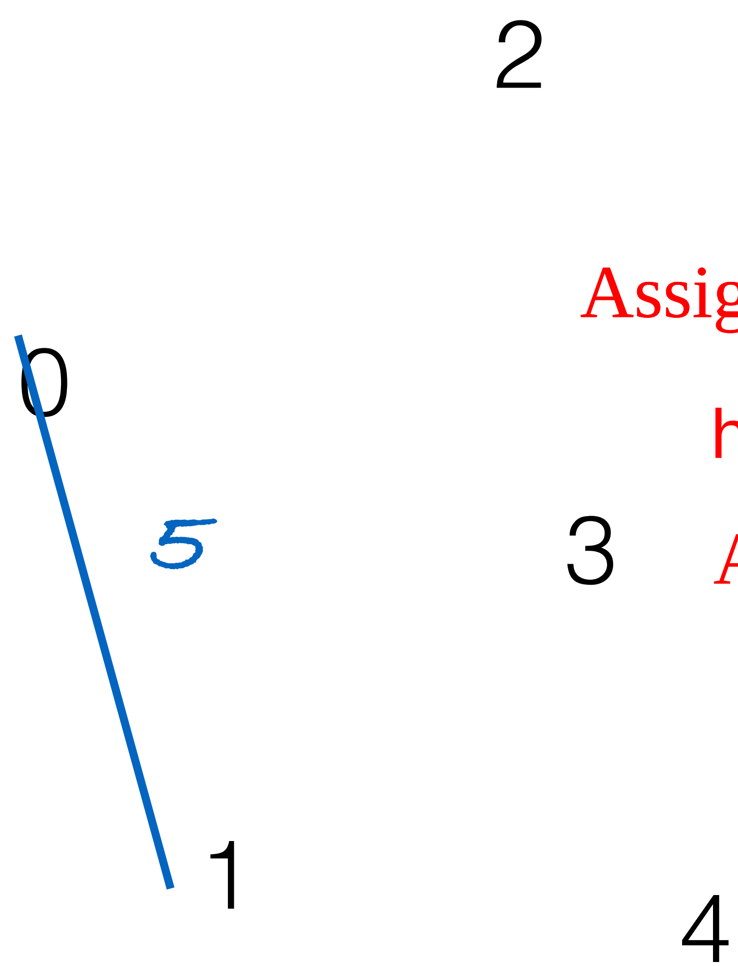
https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Closest Pair Problem (2D)

i=0   min=5



Assignment Project Exam Help

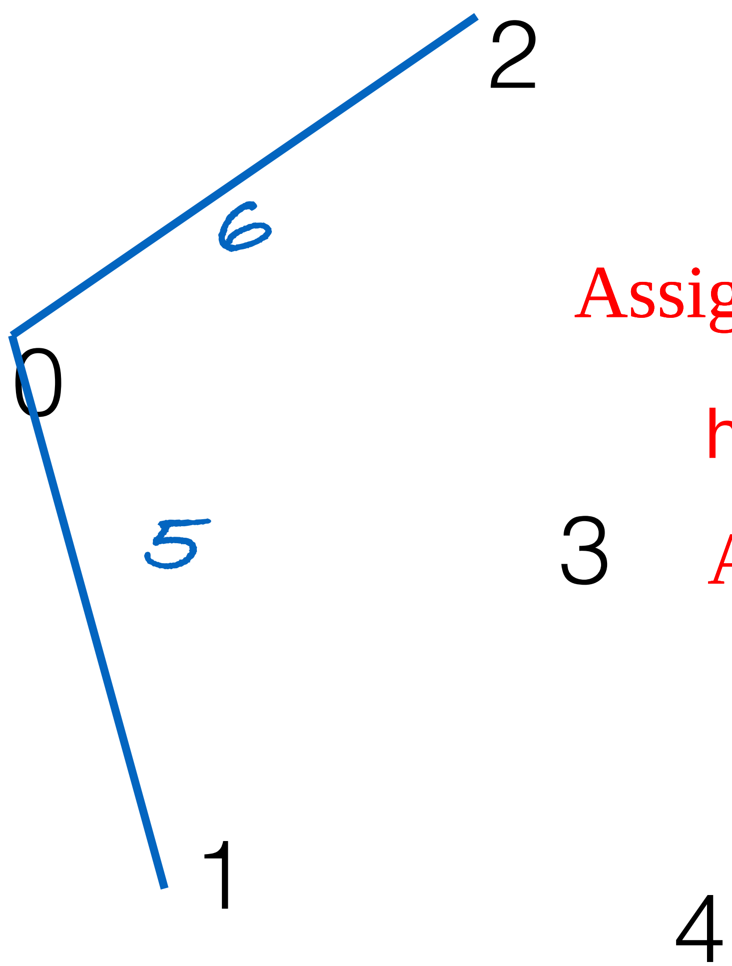
https://eduassistpro.github.io/

Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Closest Pair Problem (2D)

i=0   min=5



Assignment Project Exam Help

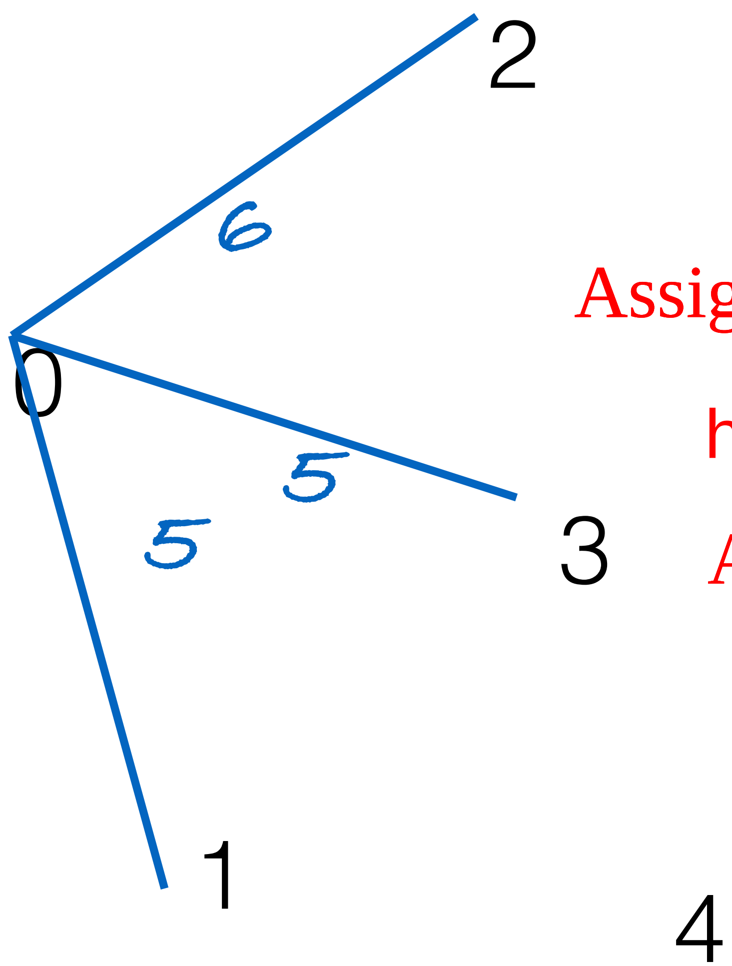
https://eduassistpro.github.io/

Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Closest Pair Problem (2D)

i=0 min=5



Assignment Project Exam Help

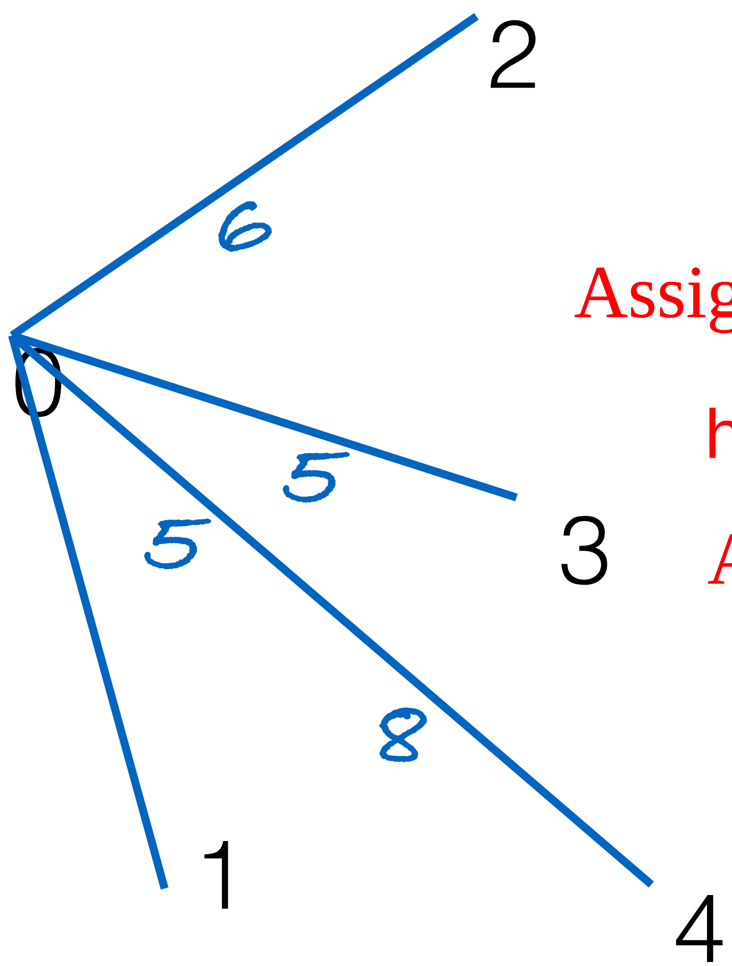
https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Closest Pair Problem (2D)

i=0 min=5



Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Closest Pair Problem (2D)

i=1 min=5

2

Assignment Project Exam Help

0

https://eduassistpro.github.io/

3 Add WeChat edu_assist_pro

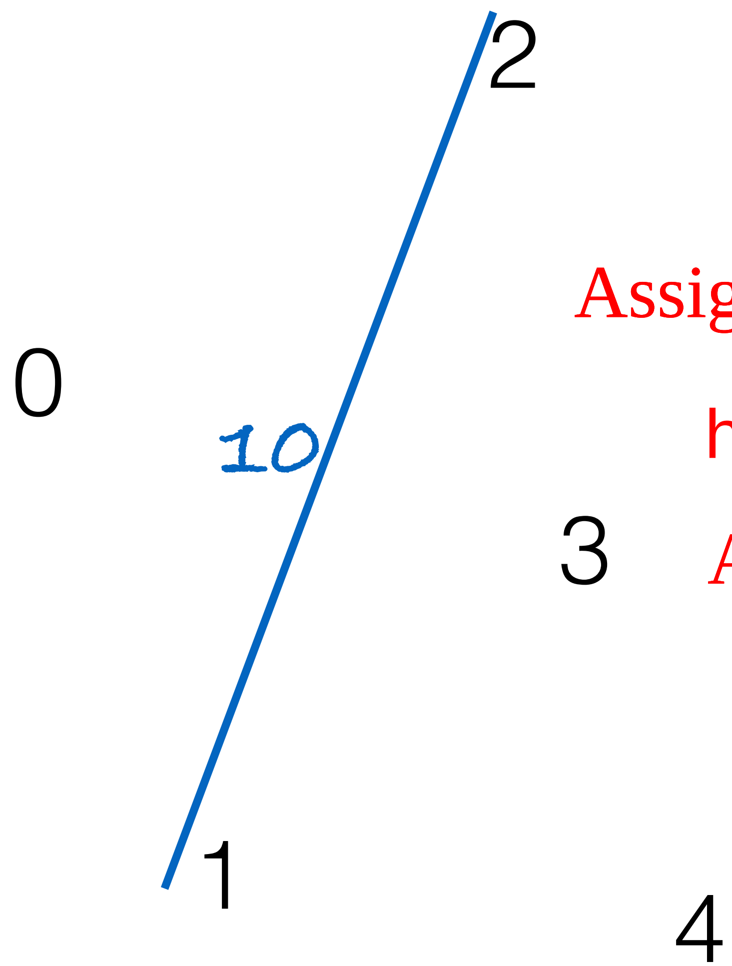
1

4

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Closest Pair Problem (2D)

i=1 min=5



Assignment Project Exam Help

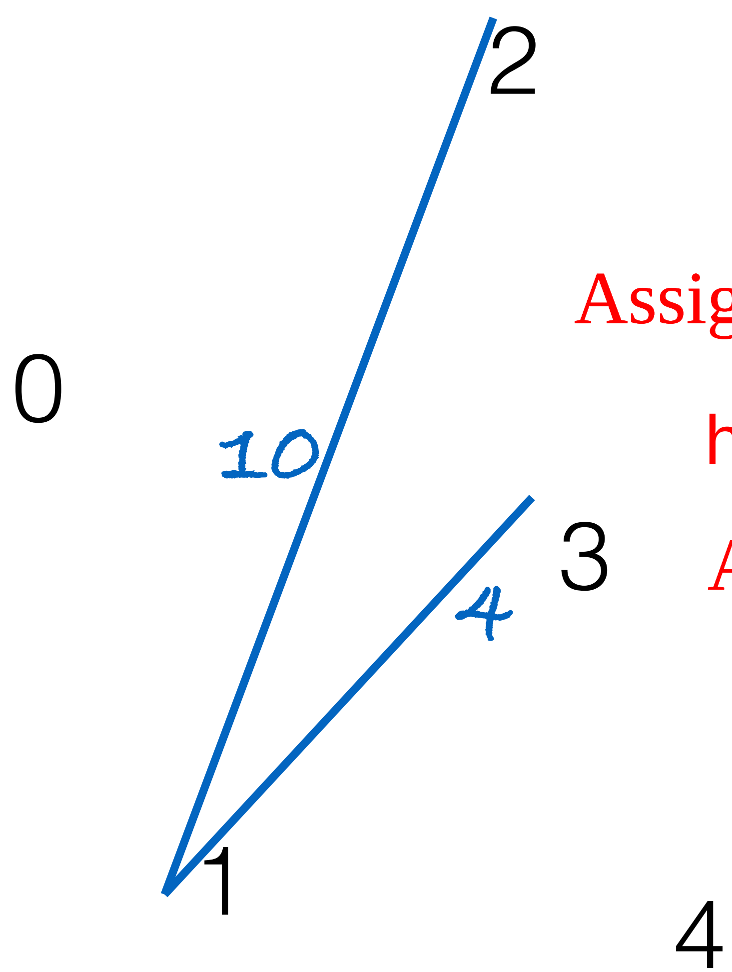
https://eduassistpro.github.io/

Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Closest Pair Problem (2D)

i=1   min=5



Assignment Project Exam Help

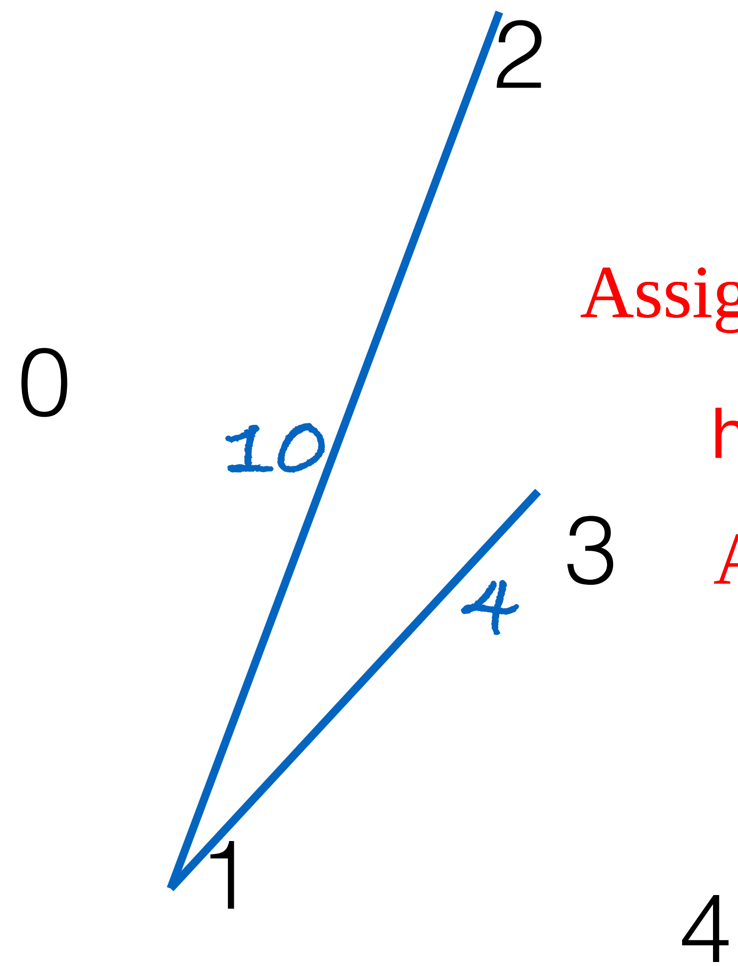
https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Closest Pair Problem (2D)

i=1 min=4



Assignment Project Exam Help

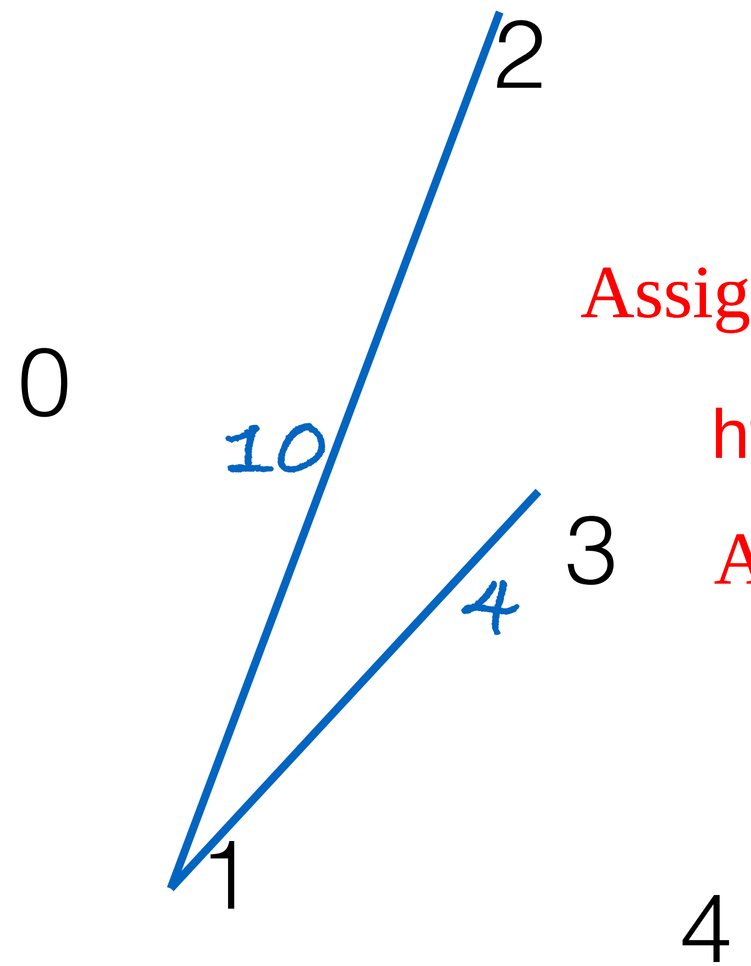
https://eduassistpro.github.io/

Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Closest Pair Problem (2D)

i=1 min=4



and so on until

3 and j=4

d the minimum here)

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Analysing Brute Force Closest Pair Algorithm

- Not hard to see that the algorithm is
- Note, however, that we can speed up the algorithm considerably, by Assignment Project Exam Help notonicity of the square root funct https://eduassistpro.github.io/

Add WeChat edu_assist_pro

- Does that contradict the claim?
- Later we will see a clever divide and conquer approach leads to a $\Theta(n \log n)$ algorithm

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Analysing Brute Force Closest Pair Algorithm

THE UNIVERSITY OF MELBOURNE

- Not hard to see that the algorithm is $\Theta(n^2)$
- Note, however, that we can speed up the algorithm considerably, by Assignment Project Exam Help notonicity of the square root funct https://eduassistpro.github.io/

Add WeChat edu_assist_pro

- Does that contradict the claim?
- Later we will see a clever divide and conquer approach leads to a $\Theta(n \log n)$ algorithm

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Analysing Brute Force Closest Pair Algorithm

- Not hard to see that the algorithm is $\Theta(n^2)$
- Note, however, that we can speed up the algorithm considerably, by Assignment Project Exam Help notonicity of the square root funct https://eduassistpro.github.io/ $\sqrt{a} < \sqrt{b}$
Add WeChat edu_assist_pro
- Does that contradict the claim?
- Later we will see a clever divide and conquer approach leads to a $\Theta(n \log n)$ algorithm

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Analysing Brute Force Closest Pair Algorithm

- Not hard to see that the algorithm is $\Theta(n^2)$

- Note, however, that we can speed up the algorithm considerably, by Assignment Project Exam Help notonicity of the square root funct https://eduassistpro.github.io/ $\sqrt{a} < \sqrt{b}$

Add WeChat edu_assist_pro

- Does that contradict the $\Theta(n^2)$ claim?

- Later we will see a clever divide and conquer approach leads to a $\Theta(n \log n)$ algorithm

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Brute Force Summary

- Simple, easy to program, widely applicable.
- Standard approach for small tasks.
- Reasonable algo problems.
- **But:** Generally inefficient—does not scale well.
- Use brute force for prototyping, or when it is known that input remains small.

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Exhaustive Search

THE UNIVERSITY OF MELBOURNE

- Problem type:
  - Combinatorial decision or optimization problems
  - Search for an element with a particular property
  - Domain grows exponentially, for example all permutations

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

- The brute-force approach—generate and test:
  - Systematically construct all possible solutions
  - Evaluate each, keeping track of the best so far
  - When all potential solutions have been examined, return the best found

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Example 1: Travelling Salesperson (TSP)

THE UNIVERSITY OF MELBOURNE

- Find the shortest **tour** (visiting each node exactly once before returning to the start) in a *weighted, undirected* graph.

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Side Note: Graph Concepts

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Side Note: Graph Concepts

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Side Note: Graph Concepts

a node

(aka vertex)

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Side Note: Graph Concepts

a node
(aka vertex)

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Side Note: Graph Concepts

THE UNIVERSITY OF MELBOURNE

a node
(aka vertex)

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

an
edge

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Side Note: Graph Concepts

THE UNIVERSITY OF MELBOURNE

a node
(aka vertex)

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

an
edge

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Side Note: Graph Concepts

an edge weight

a node (aka vertex)

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

an edge

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Side Note: Graph Concepts

an edge weight

a node (aka vertex)

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

This graph is <u>undirected</u> since edges do not have directions associated with them.

an edge

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Example 1: Travelling Salesperson (TSP)

THE UNIVERSITY OF MELBOURNE

- Find the shortest **tour** (visiting each node exactly once before returning to the start) in a *weighted, undirected* graph.

Assignment Project Exam Help

Tours

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Example 2: Knapsack

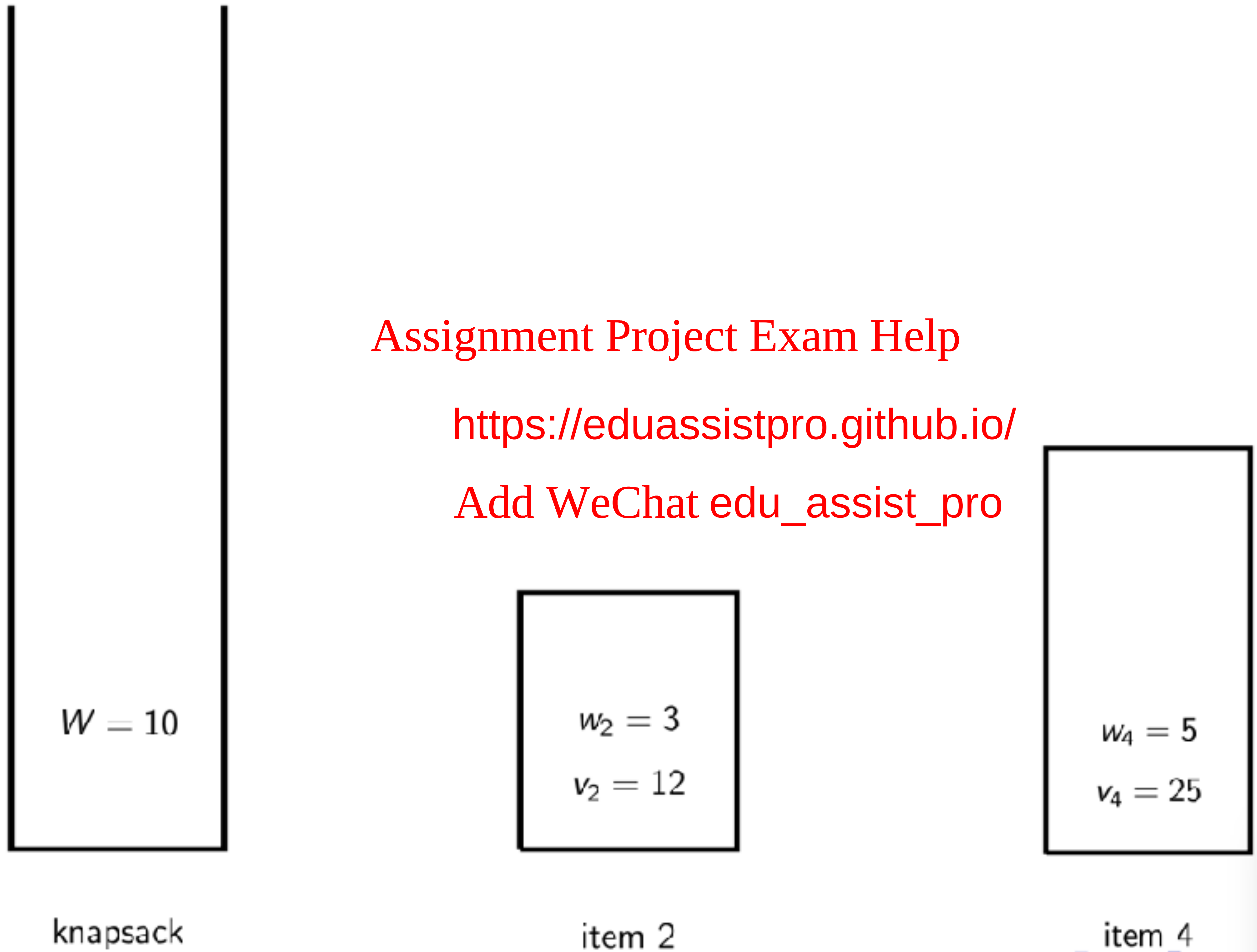
- Given *n* items with
  - Weights: w_1, w_2, …, w_n
  - Values: v_1, v_2, …, v_n
  - Knapsack of c

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

- find the most valuable selection of items whose combined weight does not exceed *W*

Copyright University of Melbourne 2016, provided under Creativ e

Knapsack Example



Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Knapsack: Example

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

NF means "not feasible": exhausts the capacity of the knapsack.

Later we'll find a better algorithm based on **dynamic programming**

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Comments on Exhaustive Search

- Exhaustive search algorithms have acceptable running times **only for very small instances.**

- In many cases there are better alternatives, for example, Eulerian tours, shortest paths, minimum spanning trees, .

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

- But for some problems, it is that there is essentially no better alternative.

- For a large class of important problems, it **appears** that there is no better alternative, but we have no proof either way.

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Hamiltonian Tours

- The Hamiltonian tour problem is this:

- In a given undirected graph, is there a simple tour (a path that visits each **node** exactly once, except it returns to the starting node)?

Assignment Project Exam Help

- Is there a Hamiltoni[illegible]ph?

https://eduassistpro.github.io/
Add WeChat edu_assist_pro

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Eulerian Tours

THE UNIVERSITY OF MELBOURNE

- The Eulerian tour problem is this:
- In a given undirected graph, is there a path which visits each **edge** exactly once?
- Is there a Euleria h?

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Hard and Easy Problems

- Recall that by a **problem** we usually mean a parametric problem: an infinite family of problem “instances”.

- Sometimes our intuition about the difficulty of problems is not very reliable. The Hamiltonian Tour problem and the Eulerian Tour problem look very similar, but one is hard and the other is easy. We shall see more exampl n later.

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

- For many important **optimization** p e do not know of solutions that are essentially better ustive search (a whole raft of **NP-complete** problems, including TSP, knapsack).

- In those cases we may look for **approximation algorithms** that are fast and still find solutions that are reasonably close to the optimal.

- We plan to return to this idea in Week 12.

Copyright University of Melbourne 2016, provided under Creative Commons Attribution License

Next Up

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

- **Recursion** as a technique

 Copyright University of Melbourne 2016, provided under Creative Commons Attribution License