

# **External Documentation**

COMP7084001 - Multimedia Systems - LC01

Even Semester Year 2021/2022

## **1. Group Member**

- Linggar Tembus Nusantara - 2440107771
- Nadya Tyandra - 2440032820
- Randy Antonio - 2440034170

## **2. Project Title**

CodeActiO(n)

## **3. Introduction**

CodeActiO(n) is a web-based application that supports Algorithm Design and Analysis course, especially for the topic about time complexity and its type (Constant Time, Linear Time, Logarithmic Time, Quadratic Time, Exponential Time, and Factorial Time). CodeActiO(n) is created using HTML, CSS, and JavaScript language. This web-based application consists of 3 pages, Home page, Tutorials page, and About Us page. CodeActiO(n) delivers the explanation about time complexity in video and article. This application also shows code snippets in C++ language as an example. This application also shows our partners and testimonies from our subscribers that were displayed in the carousel. Users could play the background music of Mario Bros Theme Song in the Home page by clicking play button and pause it by clicking pause button. If users click the Tutorial Video in Tutorials page, it will link to the YouTube Video that explains the topic.

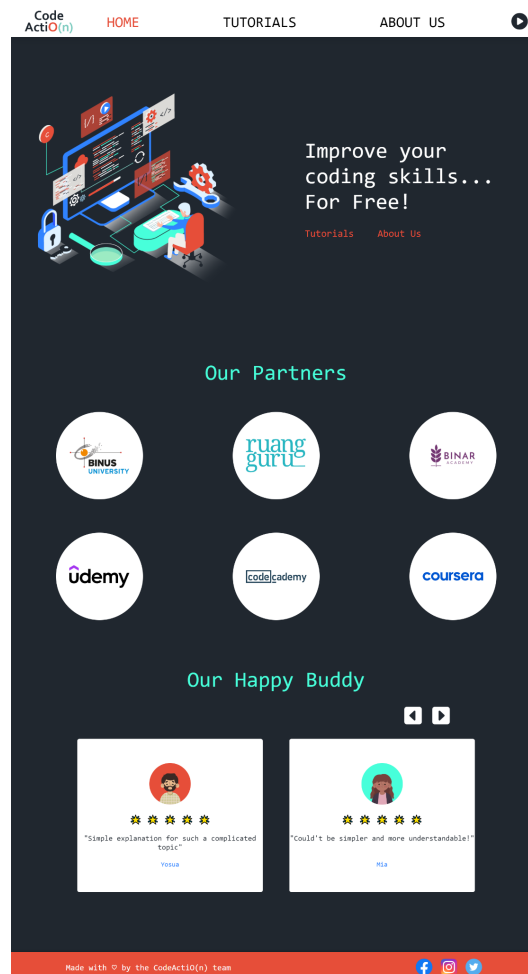
## **4. Report and Documentation**

### **a. Home**

As a landing page, this page is divided up into 3 parts: header, body, and footer.

- On the header, users will see CodeActio(n)'s logo and buttons that will direct users to the home page, tutorials page, and about us page. The page where users are currently

- at is highlighted in red color. On the right side of the header, users will also be able to play and pause background music of Mario Bros Theme Song.
- The second part is the body. On the body, it's also divided into 3 parts, which are introduction, partners, and testimonials. Introduction shows an image of a person coding and on the right side shows the links to the tutorials and about us pages. Below that, it shows CodeActio(n)'s partners: BINUS University, Ruang Guru, Binar Academy, Udemy, Codecademy, and Coursera. And the last part of the body is testimonials. It shows the subscribers that have tried our tutorials and gave their feedback on it. The testimonials were displayed in a carousel. Users can click the left arrow and right arrow to swipe and see other testimony. However, the carousel will automatically be swiped.
  - At the bottom is the footer. Here, it shows "Made with ♥ by the CodeActio(n) team" and links to our social media.



## b. Tutorials

In this page, there are 2 main parts: video and explanation & example.

- On the video part, it will show the title (“Video”) of the subpage and below it is an embed youtube link that is wrapped in a box, showing a title of the video. The video is posted on youtube and can be controlled as a normal youtube video could.
- Below that is the explanation and example part. Here, it shows the different types of time complexity, which is a part of the Algorithm and Design course. Here, the complexities are divided into 6 different time complexities, which are constant time, linear time, logarithmic time, quadratic time, exponential time, and factorial time. Every time complexity comes with an explanation and a code snippet showing a code example of that time complexity, written in C++ language.

[Code ActiO\(n\)](#) [HOME](#) [TUTORIALS](#) [ABOUT US](#)

# Video

Big O Notation - Video Tutorial

Time Complexity:  $O(n^2)$

Constant Time!

Factorial Time!

$O(n^2)$

Big O!

Time Complexity

Time complexity is the amount of time taken by an algorithm to run, as a function of the length of the input. It is not going to measure the total execution time of an algorithm. Rather, it is going to give information about the operation in terms of time when the number of operations increase or decrease in an algorithm. Time complexity measures the time taken to execute one statement of code for an algorithm. It is determined by the number of times that statement gets executed is equal to  $n$  multiplied by the time required to run that function each time.

## Explanation & Example

### 1 $O(1)$ - Constant Time

When an algorithm is not reliant on the input size  $n$ , it is said to have constant time that can perform in  $O(1)$  time. The runtime will always be the same, regardless of the input size  $n$ . For example, printing array at index 0.

```
print("Hello", a[0]);
```

### 2 $O(n)$ - Linear Time

When the runtime of an algorithm increases linearly with the length of the input, it is said to have linear time complexity. When a function checks all of the values in input data, it is said to have linear time that can perform in  $O(n)$  time. For example, finding a matching pair.

```
function findPair(a, n) {
  for (let i = 0; i < n; i++) {
    if (a[i] === 'pair') {
      return true;
    }
  }
}
```

### 3 $O(\log n)$ - Logarithmic Time

An algorithm is said to have a logarithmic time complexity when the running time increases slowly with the input size  $n$ . The time complexity is proportional to the logarithm of input size  $n$ . For example, binary search.

```
function binarySearch(a, target) {
  let low = 0, high = a.length - 1;
  while (low <= high) {
    let mid = Math.floor((low + high) / 2);
    if (a[mid] === target) {
      return mid;
    } else if (a[mid] < target) {
      low = mid + 1;
    } else {
      high = mid - 1;
    }
  }
  return -1;
}
```

### 4 $O(n^2)$ - Quadratic Time

When the execution time of an algorithm increases quadratically with the length of the input, it is said to have a quadratic time complexity. In general, nested loops require this time complexity. For example, nested loops.

```
function findPair(a, n) {
  for (let i = 0; i < n; i++) {
    for (let j = 0; j < n; j++) {
      if (a[i] === a[j]) {
        return true;
      }
    }
  }
}
```

### 5 $O(2^n)$ - Exponential Time

In exponential time, with each addition to the  $n$  input, the growth rate doubles, and the algorithm iterates across all values of the input components. The result of operations done by the time complexity doubles every time an input unit grows by one. For example, recursive Fibonacci.

```
function fibonacci(n) {
  if (n < 2) return n;
  return fibonacci(n - 1) + fibonacci(n - 2);
}
```

### 6 $O(n!)$ - Factorial Time

Factorial is the multiplication of all positive integer numbers less than itself. In factorial time, the runtime is increasing by each iteration. For example, finding all possible permutations of an array. Factorial time grows quickly, thus stay away if possible. For example, finding all possible permutations of an array.

```
function permute(a) {
  if (a.length === 0) {
    return [''];
  }
  let result = [];
  for (let i = 0; i < a.length; i++) {
    let element = a[i];
    let remaining = a.slice(0, i).concat(a.slice(i + 1, a.length));
    result = result.concat(remaining.map(permute));
  }
  return result;
}
```

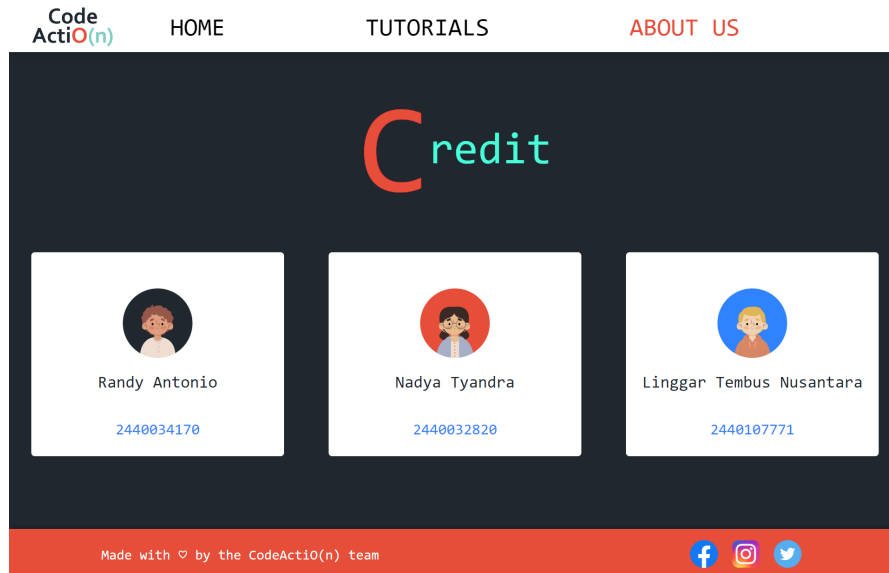
Made with by the CodeActiO(n) team

[f](#) [i](#) [t](#)

3

### c. About Us

On this About Us page, it shows the credit of the creator of this website, boxed in their separate boxes, titled “Credit”.



## 5. Hidden creativities

### - Navigation bar

Every menu has some states. This is the text color when the menu is in normal state (not hovered and not opened).

HOME

This is the text color when the menu is hovered.

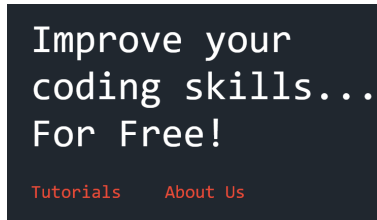
HOME

This is the text color when the menu is opened.

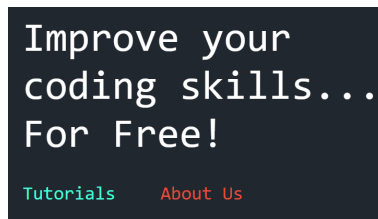
HOME

### - Introduction part

The “Tutorials” and “About Us” button in the introduction part has some states. This is the text color when the button is in normal state (not hovered and not opened).



This is the text color when the “Tutorials” button is hovered.



#### - **Play and pause button**

In the Home page, users can play the background music of Mario Bros Theme Song by clicking the play button. When the button is hovered, it will show a slightly lowered opacity (dimmer).



Users can pause the background music by clicking the pause button.

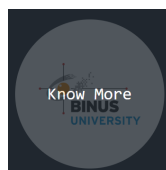


#### - **Partners button**

Each partner button has states. This is their logo when in normal state (not hovered).

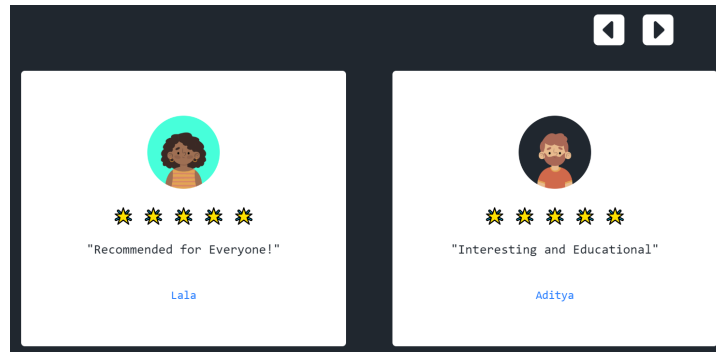


This is their logo when hovered, the logo becomes darker and “Know More” text occurs. If the user clicks the logo, the new tab of the partner’s official website is opened.



- **Testimony carousel**

Users could click the left arrow to swipe to the left card and click the right arrow to swipe to the right card. However, the carousel will automatically be swiped.



- **Social media button**

If users click the social media button, it will open the new tab of social media's official website.



- **Video Tutorial**

If users click the video tutorial, the application will play a video from YouTube about time complexity.



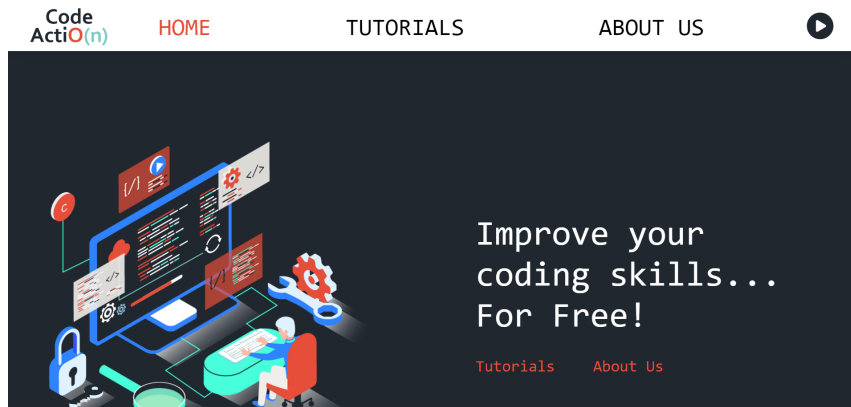
- **Logo**

If users click CodeActiO(n)'s logo in each page, the application will link to the Home page.



## - Scroll bar

The scroll bar is custom made in blue and white.



## 6. Reference

### a. Text

- Typeface: Consolas
- Size: 16 px, 18px, 20px, 22 px, 24 px, 30 px, 36 px, 40 px, 48 px, 52 px, 60 px, 64 px, 184 px, 200 px
- Font:
  - ❖ Consolas 16 px regular
  - ❖ Consolas 18 px regular
  - ❖ Consolas 20 px regular
  - ❖ Consolas 22 px regular
  - ❖ Consolas 24 px regular
  - ❖ Consolas 30 px regular
  - ❖ Consolas 36 px regular
  - ❖ Consolas 40 px regular
  - ❖ Consolas 48 px regular
  - ❖ Consolas 52 px regular
  - ❖ Consolas 60 px regular
  - ❖ Consolas 64 px regular
  - ❖ Consolas 184 px regular
  - ❖ Consolas 200 px regular

## **b. Color**

- Black: #20272F
- Red: #E64E39
- Tosca: #47FFDA
- White: #FFFFFF
- Blue: #3083FF
- Sunset Orange: #FF605C
- Pastel Orange: #FFBD44
- Malachite: #00CA4E

## **c. Content**

- Introduction, Constant time, Linear time, and Quadratic time:  
<https://www.mygreatlearning.com/blog/why-is-time-complexity-essential/#:~:text=Time%20complexity%20is%20the%20amount,you%20understand%20time%20complexity%20clearly>
- Logarithmic time:  
<https://learn2torials.com/a/logarithmic-time-complexity>
- Exponential time:  
<https://towardsdatascience.com/essential-programming-time-complexity-a95bb2608cac#:~:text=Exponential%20Time%20Complexity%3A%20O>
- Exponential time example (recursive Fibonacci):  
<https://www.geeksforgeeks.org/program-for-nth-fibonacci-number/>
- Factorial time:  
<https://adrianmejia.com/most-popular-algorithms-time-complexity-every-programmer-should-know-free-online-tutorial-course/#All-running-complexities-graphs>

## **d. Image**

- Binar Academy logo: <https://www.binaracademy.com/>
- Ruangguru logo: <https://www.ruangguru.com/>
- Binus logo: <http://binus.ac.id/>
- Udemy logo: <https://www.udemy.com/>



- Coursera logo: <https://www.coursera.org/>
- Codecademy logo: <https://www.codecademy.com/>
- Twitter logo: <http://www.twitter.com/>
- Instagram logo: <http://www.instagram.com/>
- Facebook logo: <http://www.facebook.com/>
- Code snippet image maker (Seti theme and C++ language): <https://carbon.now.sh/>
- Isometric programming image at homepage:  
[https://www.freepik.com/free-vector/isometric-programming-landing-page-template\\_5071151.htm#query=coding%20isometric&position=3&from\\_view=search](https://www.freepik.com/free-vector/isometric-programming-landing-page-template_5071151.htm#query=coding%20isometric&position=3&from_view=search)

**e. Illustration**

Testimony and About Us profile icon:

[https://www.freepik.com/free-vector/hand-drawn-different-profile-icons-pack\\_17863145.htm](https://www.freepik.com/free-vector/hand-drawn-different-profile-icons-pack_17863145.htm)

**f. Video**

Tutorial Video:

<https://youtu.be/fPAmZSGkPAQ>

**g. Audio**

Super Mario Bros Theme Song:

<https://play.nintendo.com/printables/uncategorized/exclusive-download-super-mario-bros-song/>