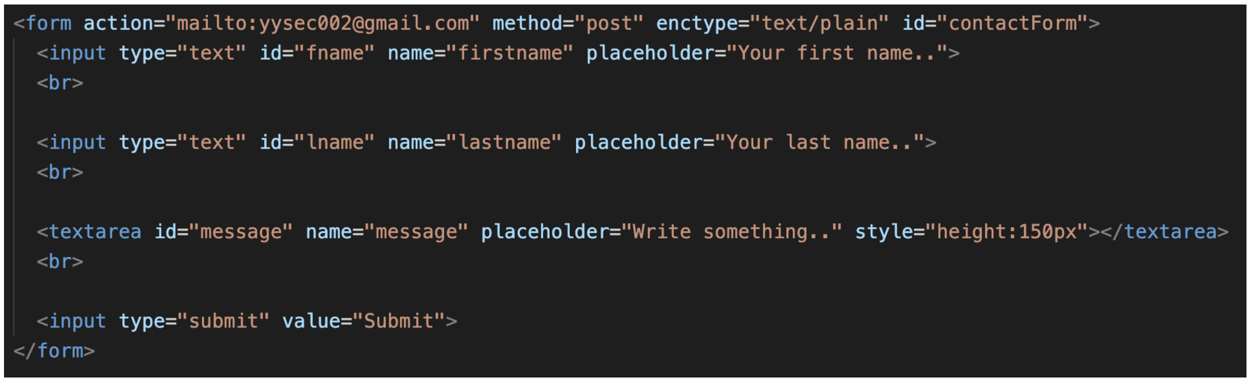
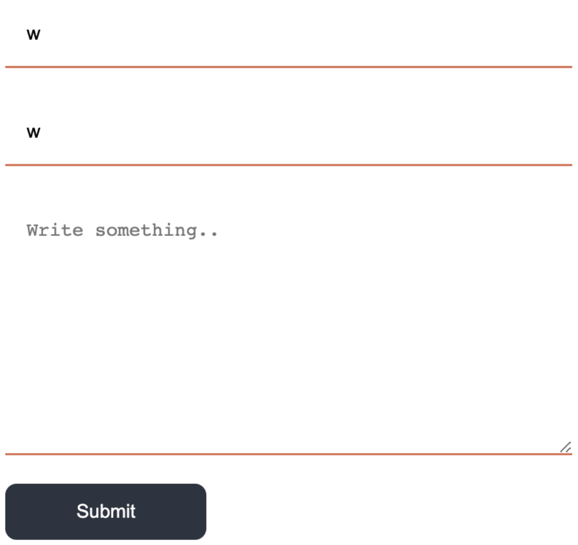
1. Email submission: the old code does check the text input. The new code uses javascript to check if the text box is empty. If any box is empty, then the email won’t be sent, and the empty text box will be marked yellow.
    a. old code snippet



```html
<form action="mailto:yysec002@gmail.com" method="post" enctype="text/plain" id="contactForm">
  <input type="text" id="fname" name="firstname" placeholder="Your first name..">
  <br>

  <input type="text" id="lname" name="lastname" placeholder="Your last name..">
  <br>

  <textarea id="message" name="message" placeholder="Write something.." style="height:150px"></textarea>
  <br>

  <input type="submit" value="Submit">
</form>
```

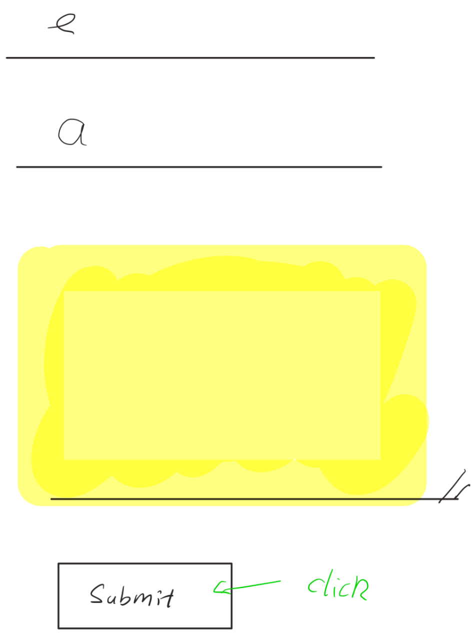
    b. old outcome



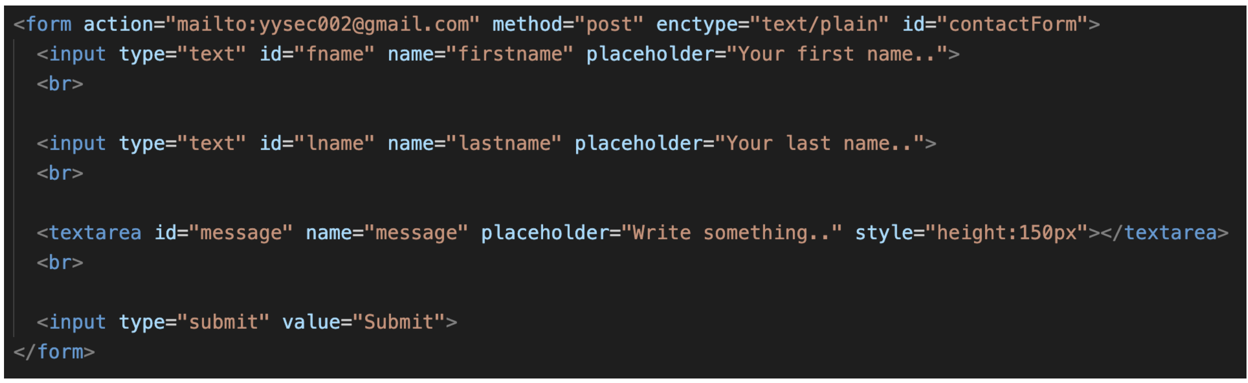
w

w

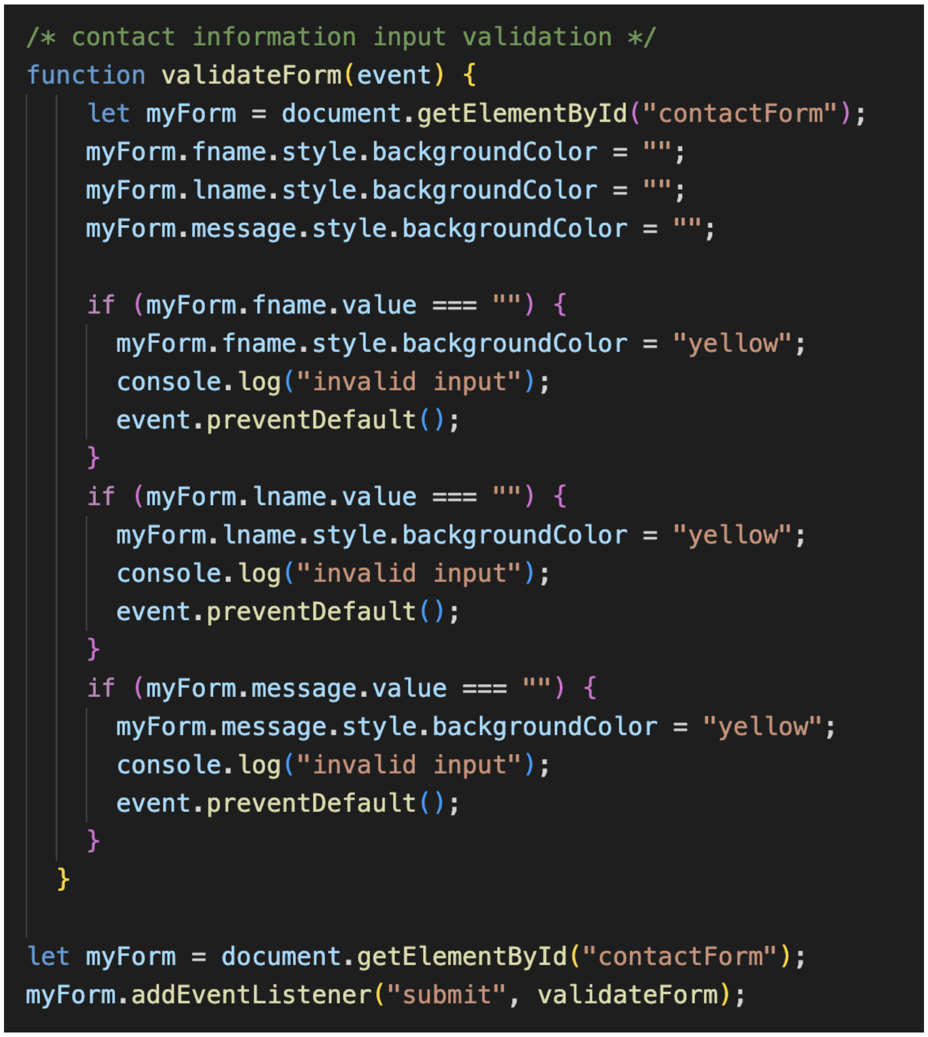
    c. desired outcome



d. new code snippet

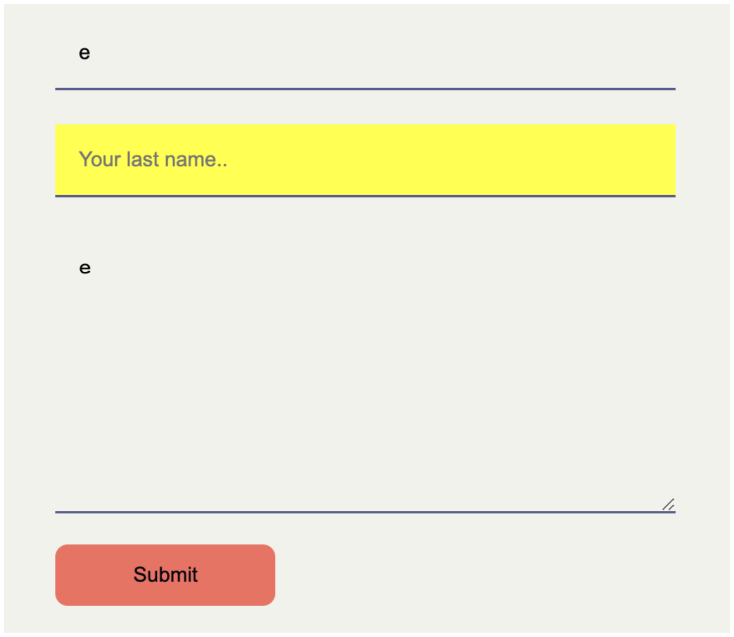


```html
<form action="mailto:yysec002@gmail.com" method="post" enctype="text/plain" id="contactForm">
  <input type="text" id="fname" name="firstname" placeholder="Your first name..">
  <br>

  <input type="text" id="lname" name="lastname" placeholder="Your last name..">
  <br>

  <textarea id="message" name="message" placeholder="Write something.." style="height:150px"></textarea>
  <br>

  <input type="submit" value="Submit">
</form>
```



```
/* contact information input validation */
function validateForm(event) {
    let myForm = document.getElementById("contactForm");
    myForm.fname.style.backgroundColor = "";
    myForm.lname.style.backgroundColor = "";
    myForm.message.style.backgroundColor = "";

    if (myForm.fname.value === "") {
      myForm.fname.style.backgroundColor = "yellow";
      console.log("invalid input");
      event.preventDefault();
    }
    if (myForm.lname.value === "") {
      myForm.lname.style.backgroundColor = "yellow";
      console.log("invalid input");
      event.preventDefault();
    }
    if (myForm.message.value === "") {
      myForm.message.style.backgroundColor = "yellow";
      console.log("invalid input");
      event.preventDefault();
    }
  }

let myForm = document.getElementById("contactForm");
myForm.addEventListener("submit", validateForm);
```

e. new outcome



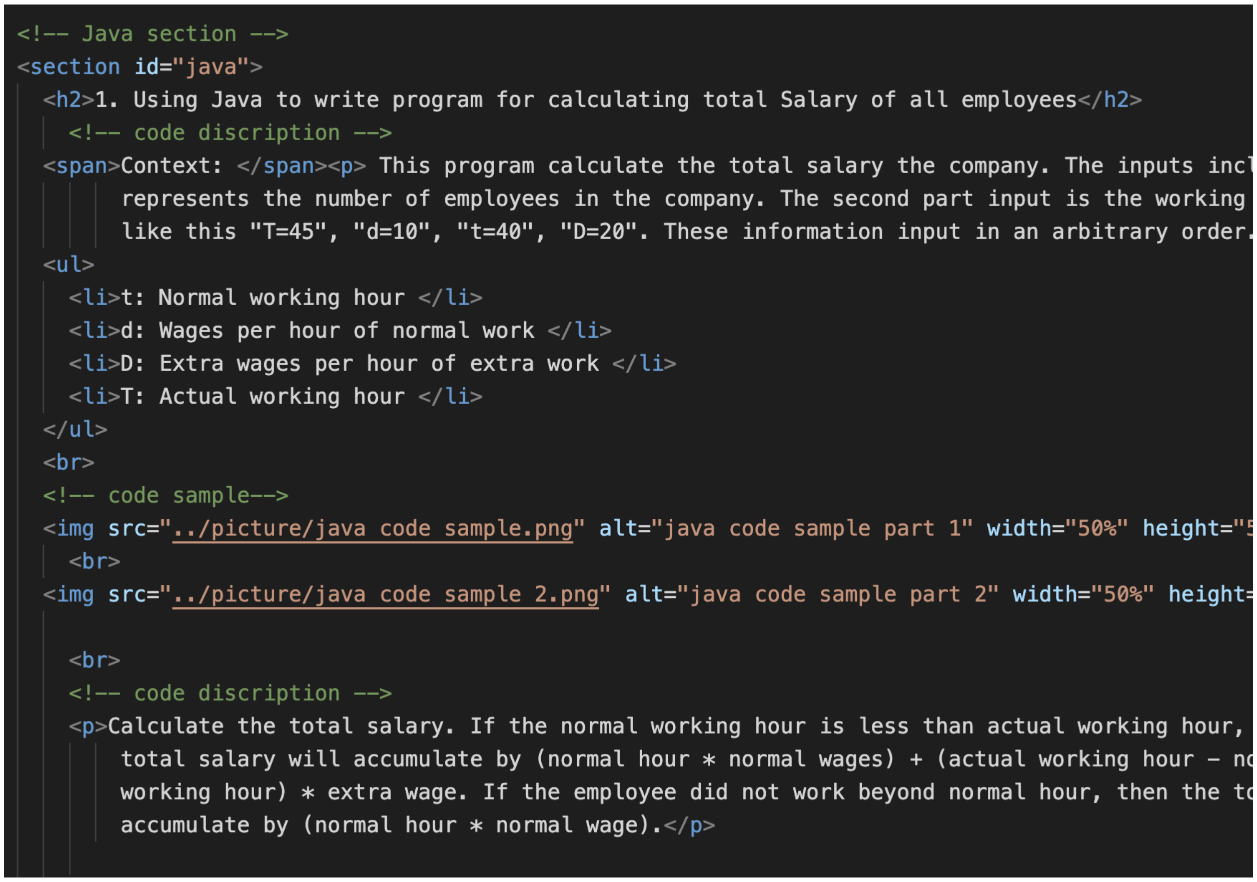
e

Your last name..

e

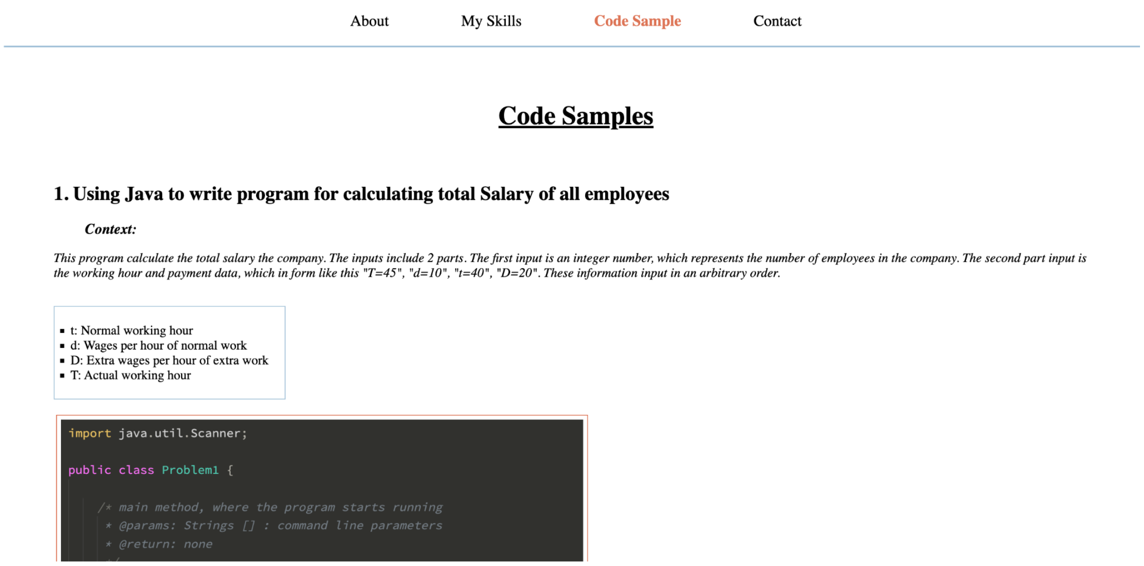
Submit

2. Code sample choice: there are two buttons, each linked to one of the code samples. The code sample will show if one of the buttons has been clicked.
    a. old code snippet



```html
<!-- Java section -->
<section id="java">
  <h2>1. Using Java to write program for calculating total Salary of all employees</h2>
    <!-- code discription -->
  <span>Context: </span><p> This program calculate the total salary the company. The inputs incl
        represents the number of employees in the company. The second part input is the working
        like this "T=45", "d=10", "t=40", "D=20". These information input in an arbitrary order.
  <ul>
    <li>t: Normal working hour </li>
    <li>d: Wages per hour of normal work </li>
    <li>D: Extra wages per hour of extra work </li>
    <li>T: Actual working hour </li>
  </ul>
  <br>
  <!-- code sample-->
  <img src="../picture/java code sample.png" alt="java code sample part 1" width="50%" height="5
    <br>
  <img src="../picture/java code sample 2.png" alt="java code sample part 2" width="50%" height=

    <br>
    <!-- code discription -->
    <p>Calculate the total salary. If the normal working hour is less than actual working hour,
        total salary will accumulate by (normal hour * normal wages) + (actual working hour - no
        working hour) * extra wage. If the employee did not work beyond normal hour, then the to
        accumulate by (normal hour * normal wage).</p>
```

    b. old outcome



Code Samples

1. Using Java to write program for calculating total Salary of all employees

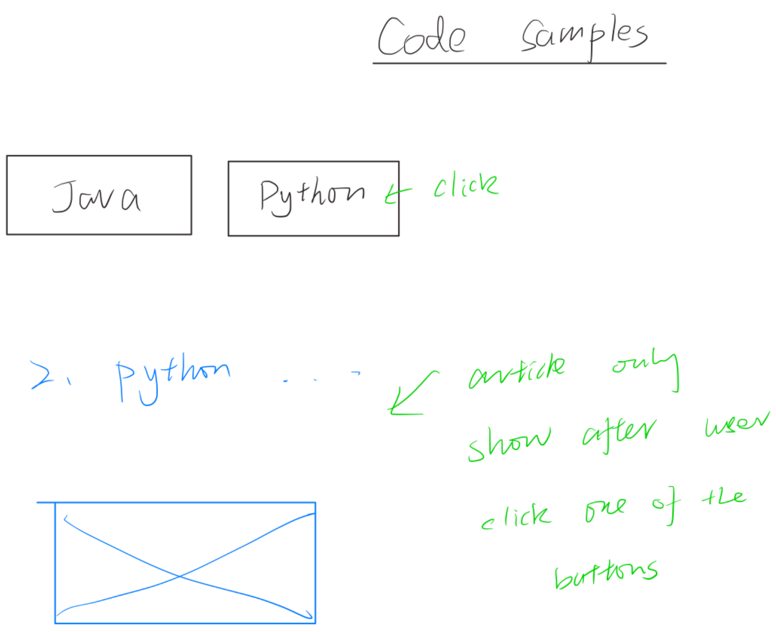
Context:

This program calculate the total salary the company. The inputs include 2 parts. The first input is an integer number, which represents the number of employees in the company. The second part input is the working hour and payment data, which in form like this "T=45", "d=10", "t=40", "D=20". These information input in an arbitrary order.

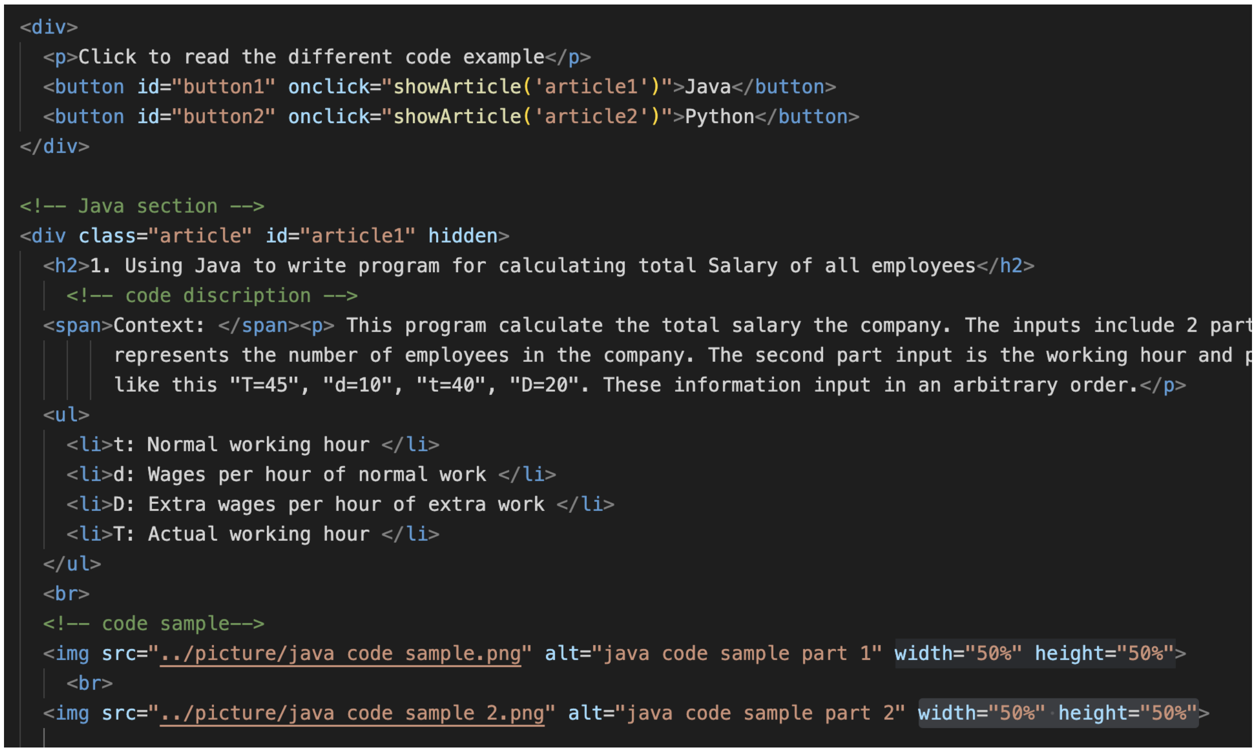
- t: Normal working hour
- d: Wages per hour of normal work
- D: Extra wages per hour of extra work
- T: Actual working hour

```java
import java.util.Scanner;

public class Problem1 {

    /* main method, where the program starts running
     * @params: Strings [] : command line parameters
     * @return: none
```

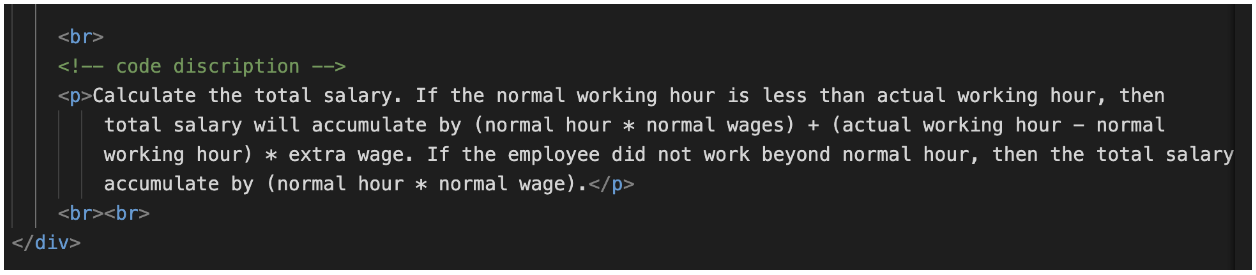
    c. desired outcome



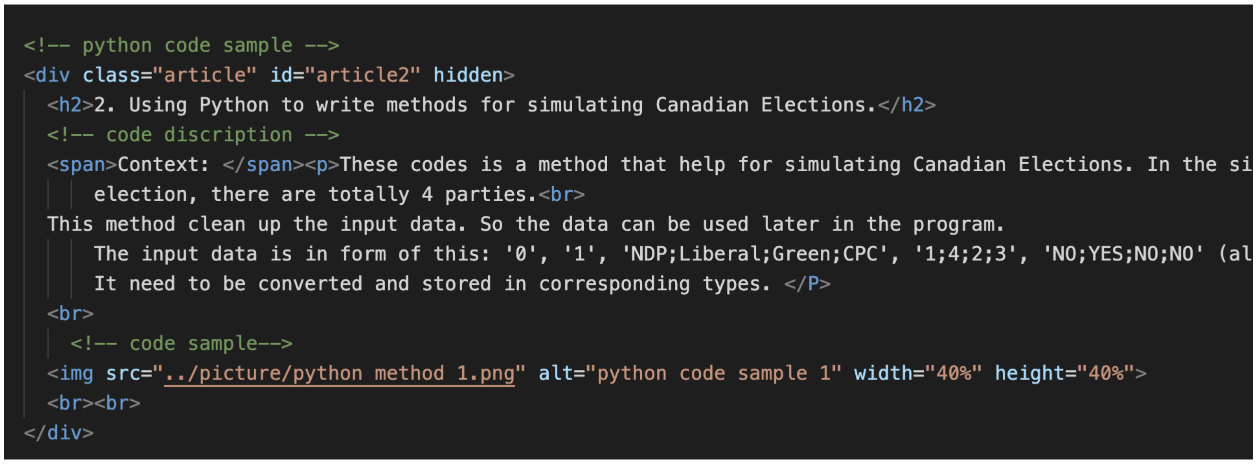
d. new code snippet



```
<div>
  <p>Click to read the different code example</p>
  <button id="button1" onclick="showArticle('article1')">Java</button>
  <button id="button2" onclick="showArticle('article2')">Python</button>
</div>

<!-- Java section -->
<div class="article" id="article1" hidden>
  <h2>1. Using Java to write program for calculating total Salary of all employees</h2>
    <!-- code discription -->
  <span>Context: </span><p> This program calculate the total salary the company. The inputs include 2 part
        represents the number of employees in the company. The second part input is the working hour and p
        like this "T=45", "d=10", "t=40", "D=20". These information input in an arbitrary order.</p>
  <ul>
    <li>t: Normal working hour </li>
    <li>d: Wages per hour of normal work </li>
    <li>D: Extra wages per hour of extra work </li>
    <li>T: Actual working hour </li>
  </ul>
  <br>
  <!-- code sample-->
  <img src="../picture/java code sample.png" alt="java code sample part 1" width="50%" height="50%">
    <br>
  <img src="../picture/java code sample 2.png" alt="java code sample part 2" width="50%" height="50%">
```

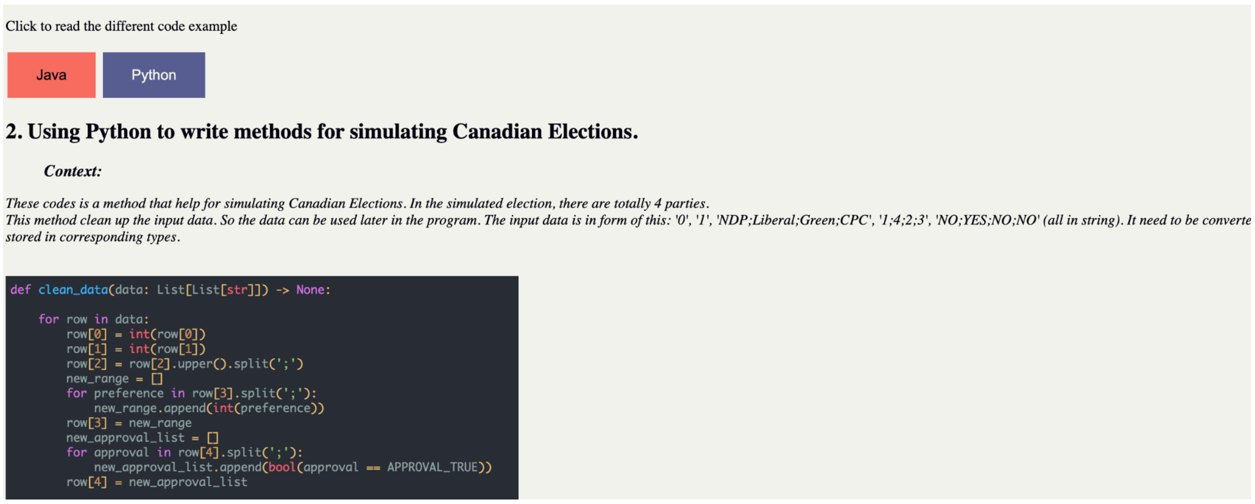


```html
    <br>
    <!-- code discription -->
    <p>Calculate the total salary. If the normal working hour is less than actual working hour, then
        total salary will accumulate by (normal hour * normal wages) + (actual working hour - normal
        working hour) * extra wage. If the employee did not work beyond normal hour, then the total salary
        accumulate by (normal hour * normal wage).</p>
    <br><br>
</div>
```



```html
<!-- python code sample -->
<div class="article" id="article2" hidden>
  <h2>2. Using Python to write methods for simulating Canadian Elections.</h2>
  <!-- code discription -->
  <span>Context: </span><p>These codes is a method that help for simulating Canadian Elections. In the si
      election, there are totally 4 parties.<br>
  This method clean up the input data. So the data can be used later in the program.
      The input data is in form of this: '0', '1', 'NDP;Liberal;Green;CPC', '1;4;2;3', 'NO;YES;NO;NO' (al
      It need to be converted and stored in corresponding types. </P>
  <br>
    <!-- code sample-->
  <img src="../picture/python method 1.png" alt="python code sample 1" width="40%" height="40%">
  <br><br>
</div>
```

e. new outcome



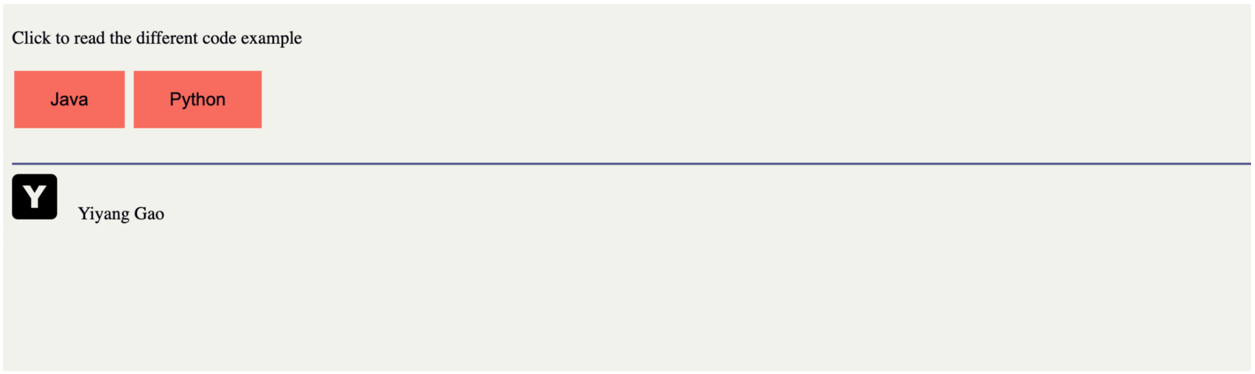
Click to read the different code example

2. Using Python to write methods for simulating Canadian Elections.

Context:

These codes is a method that help for simulating Canadian Elections. In the simulated election, there are totally 4 parties.
This method clean up the input data. So the data can be used later in the program. The input data is in form of this: '0', '1', 'NDP;Liberal;Green;CPC', '1;4;2;3', 'NO;YES;NO;NO' (all in string). It need to be converte stored in corresponding types.

```python
def clean_data(data: List[List[str]]) -> None:

    for row in data:
        row[0] = int(row[0])
        row[1] = int(row[1])
        row[2] = row[2].upper().split(';')
        new_range = []
        for preference in row[3].split(';'):
            new_range.append(int(preference))
        row[3] = new_range
        new_approval_list = []
        for approval in row[4].split(';'):
            new_approval_list.append(bool(approval == APPROVAL_TRUE))
        row[4] = new_approval_list
```



Click to read the different code example

Yiyang Gao