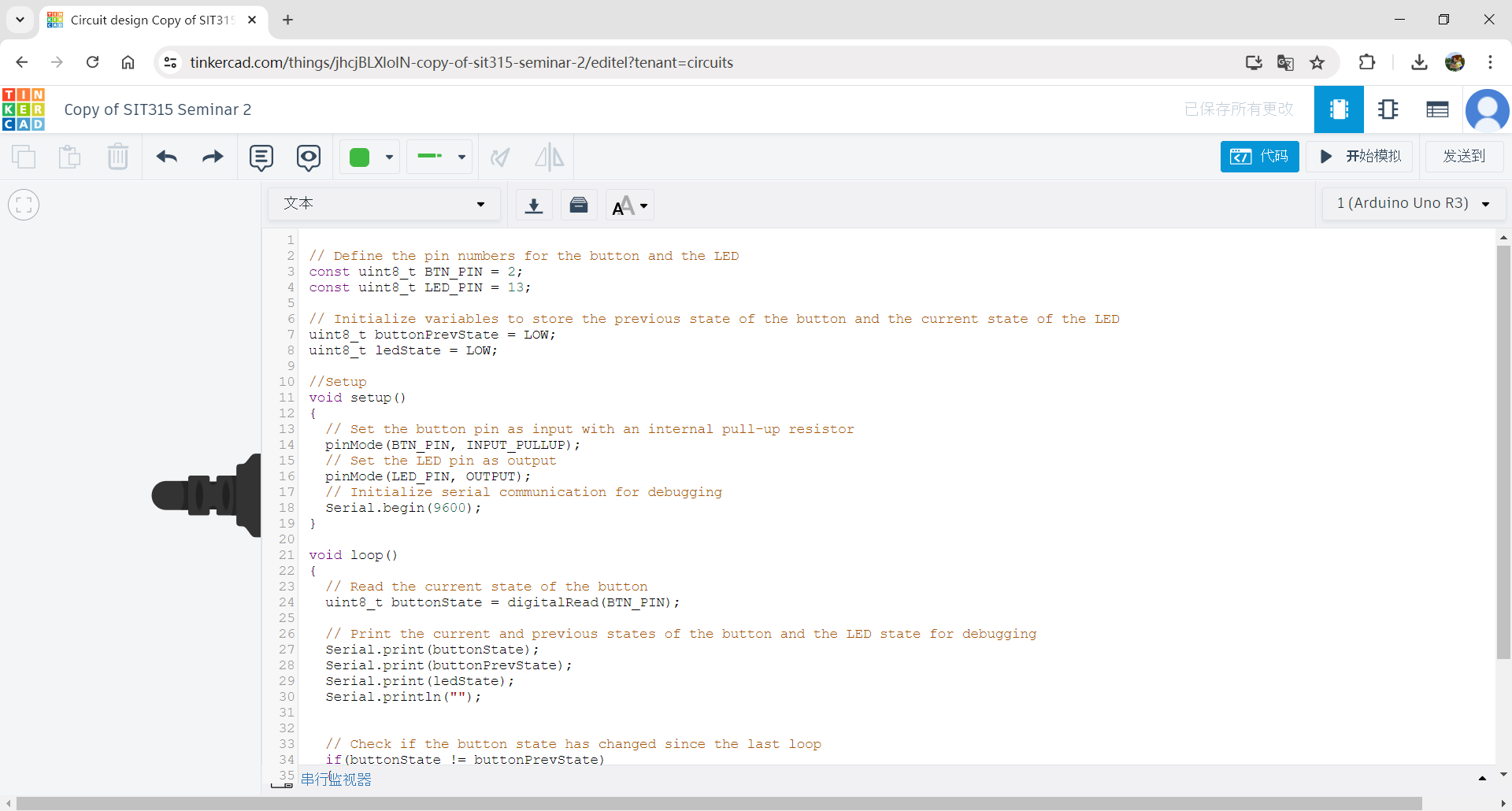
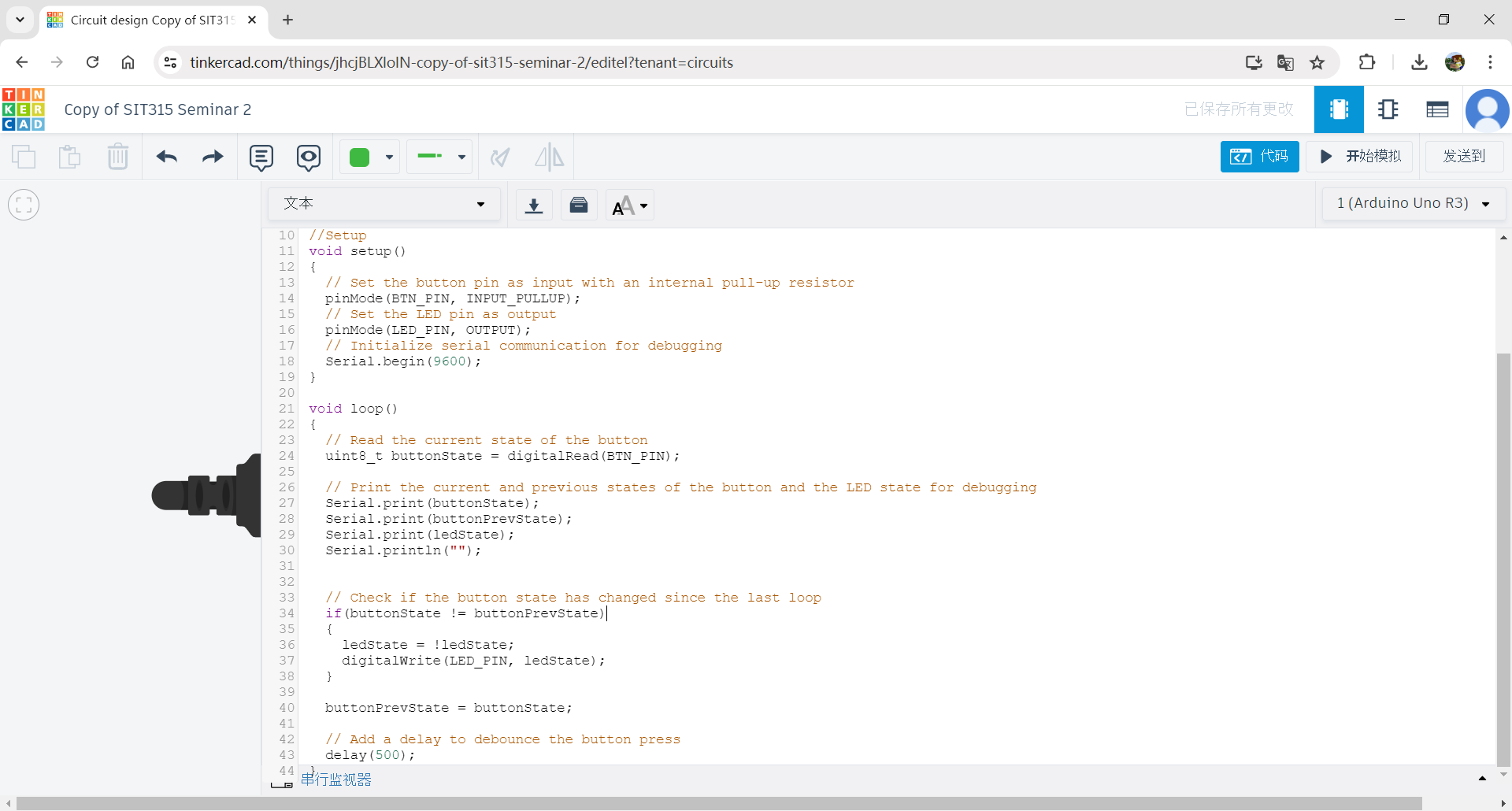
**Activity 1**

1. **Text Segment:** Stores the executable code of the program, including all functions and instructions.
2. **Data Segment:** Stores the global variable size and the static variable result from the sum function.
3. **Heap Segment:** When line 12 is executed for the third time, the heap stores the first three user input values (e.g., [4, 5, 8, 0]).   
   **Stack Segment:** The stack stores local variables p, i, and total. At this point, i=2 and total contains the sum of the first three inputs.
4. **Heap Segment:** When line 33 is executed, the dynamically allocated array on the heap has been released.   
   **Stack Segment:** The stack stores local variables p, i, and total. At this point, total contains the sum of all inputs, and i=4.

**Activity 2**

**1.**

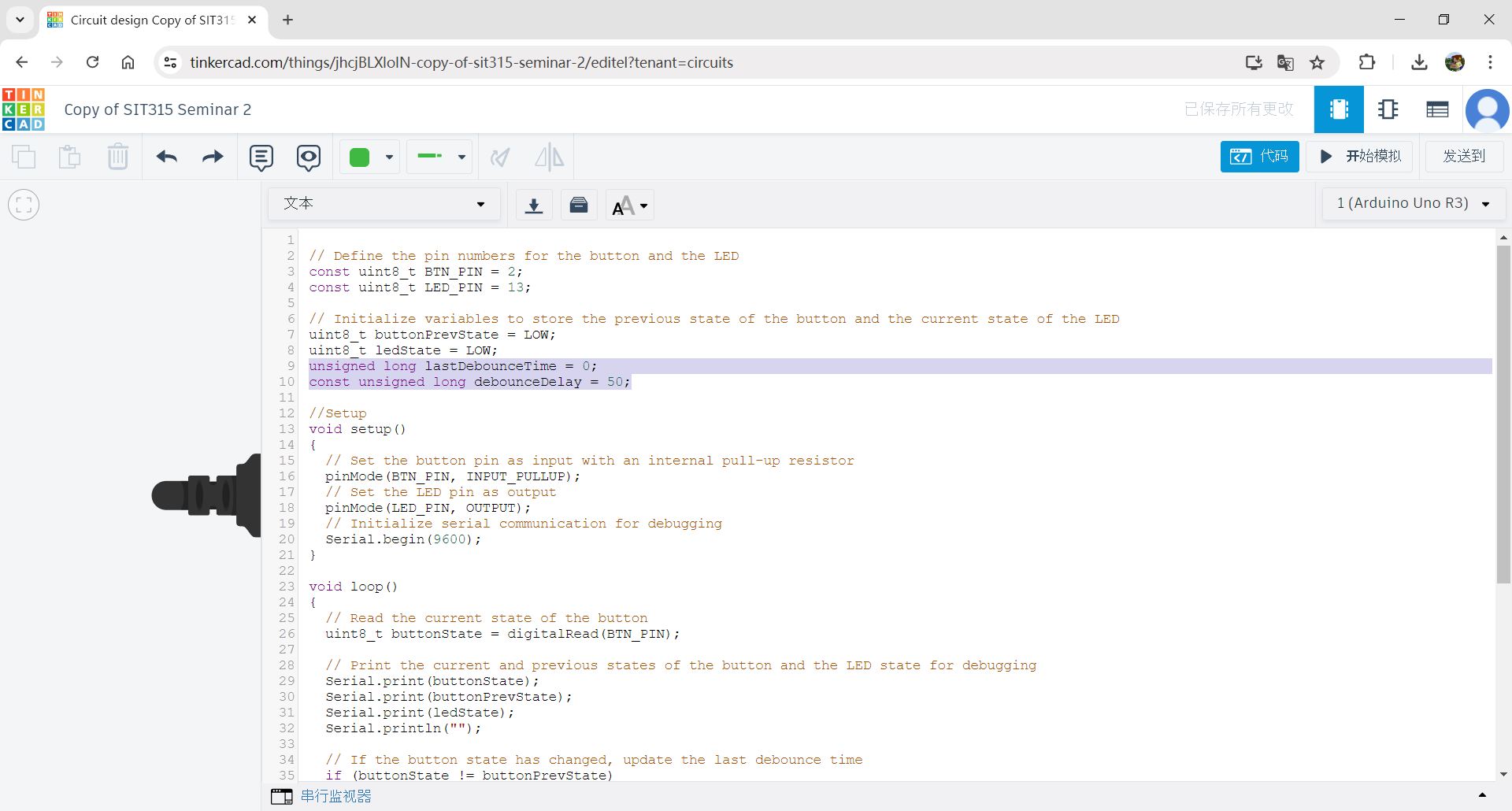


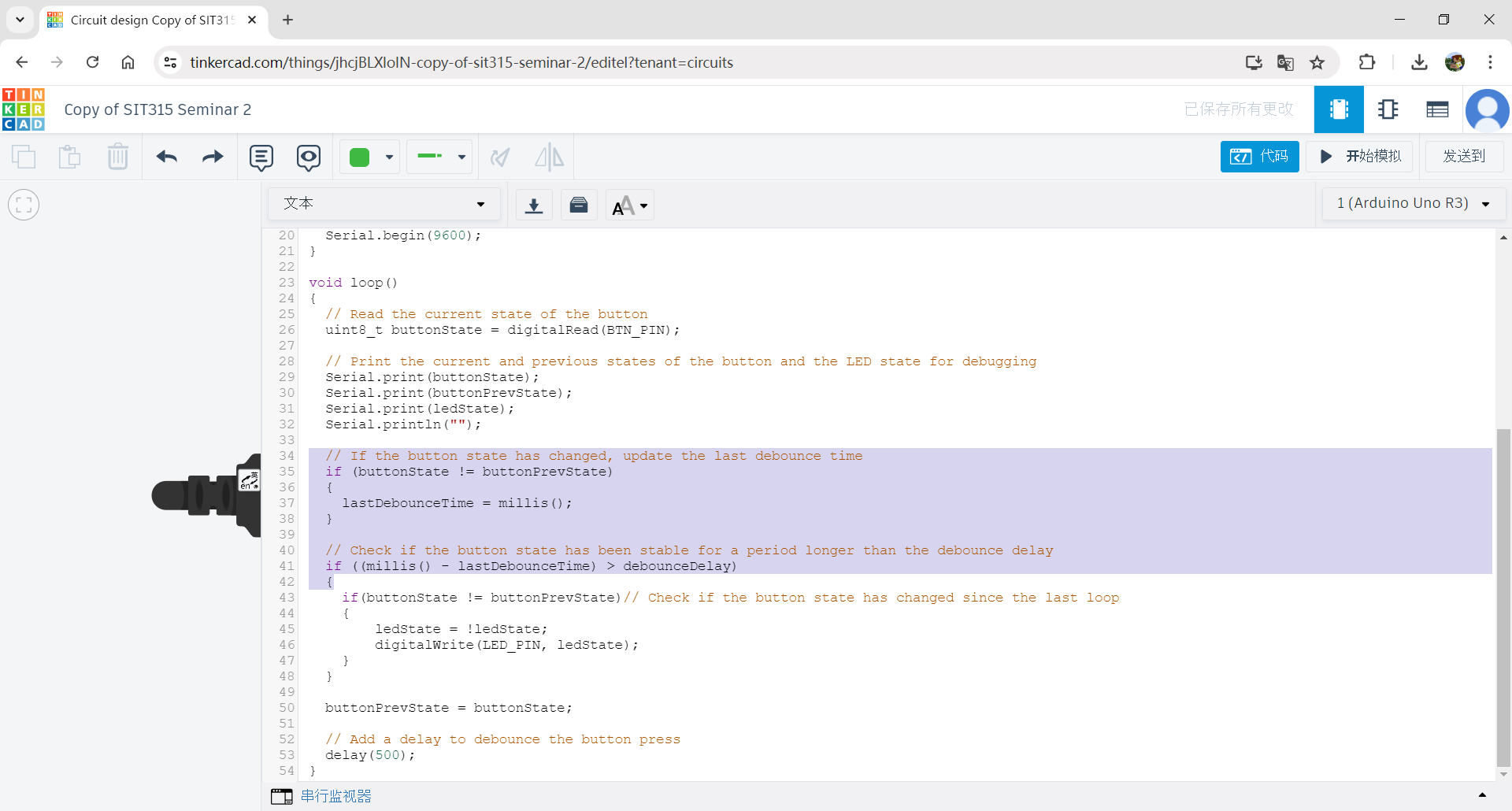


**2.**

The main problem with the code is the lack of proper debouncing for the button press, which could result in multiple toggles of the LED for a single button press due to the mechanical noise of the button.

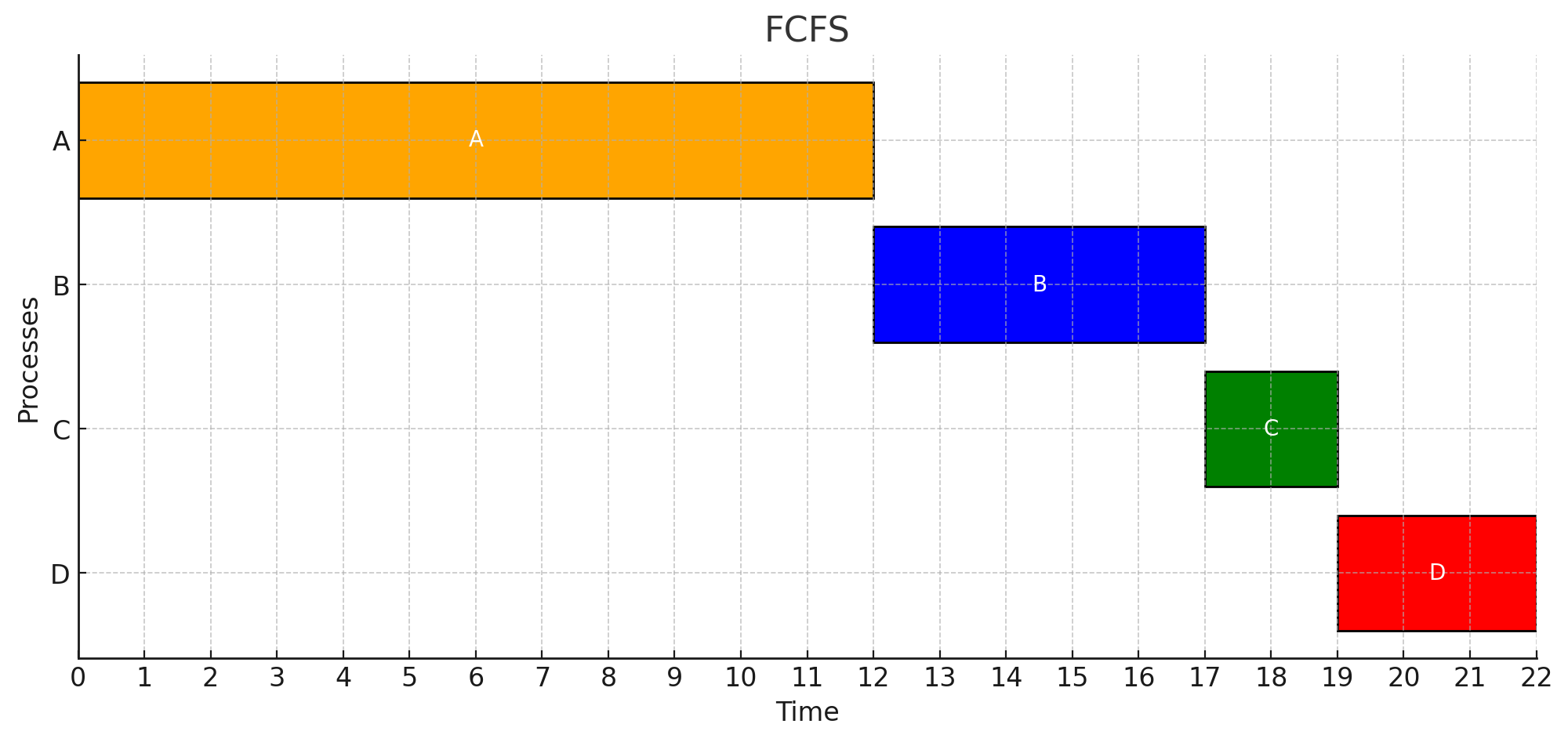
**3.**



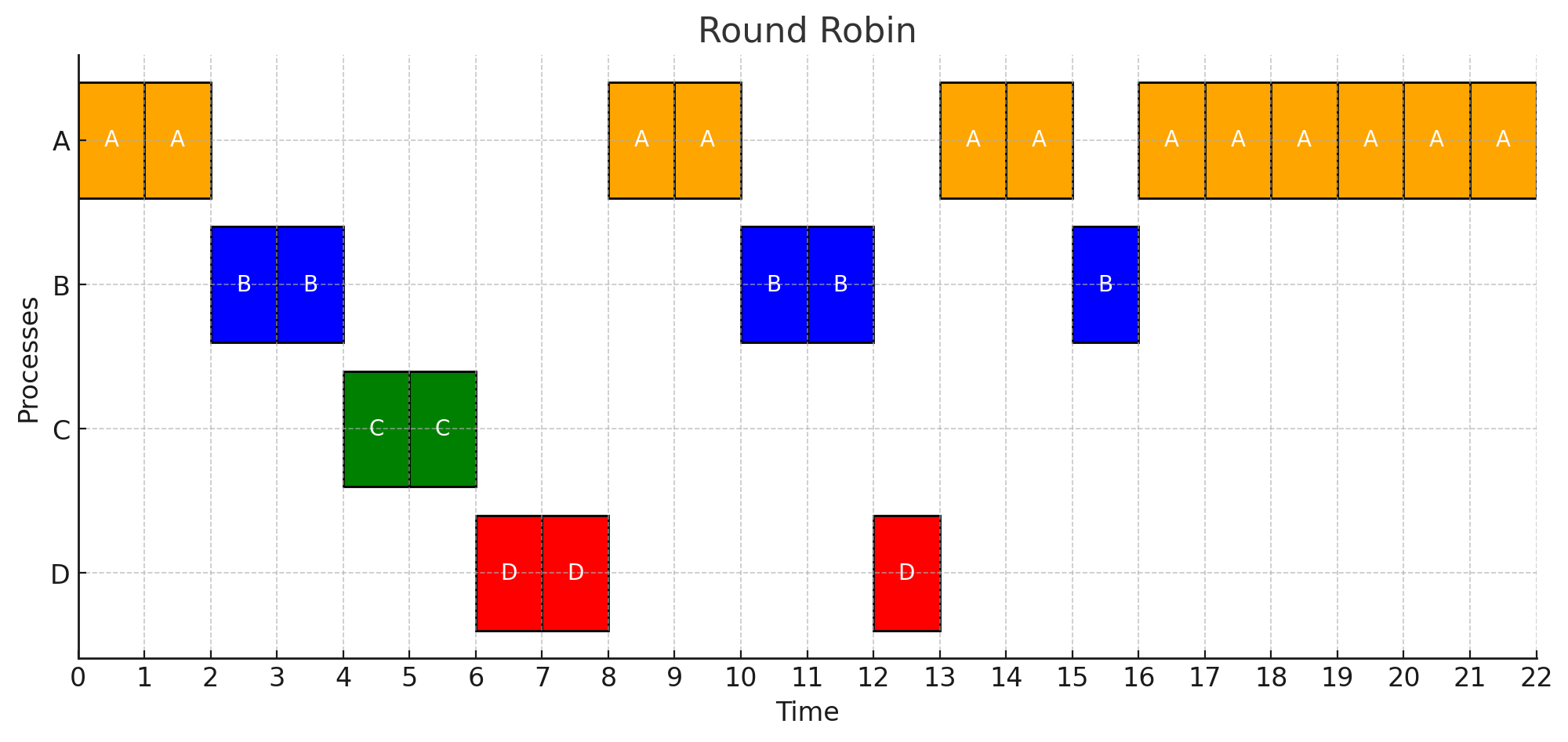


**Activity 3**

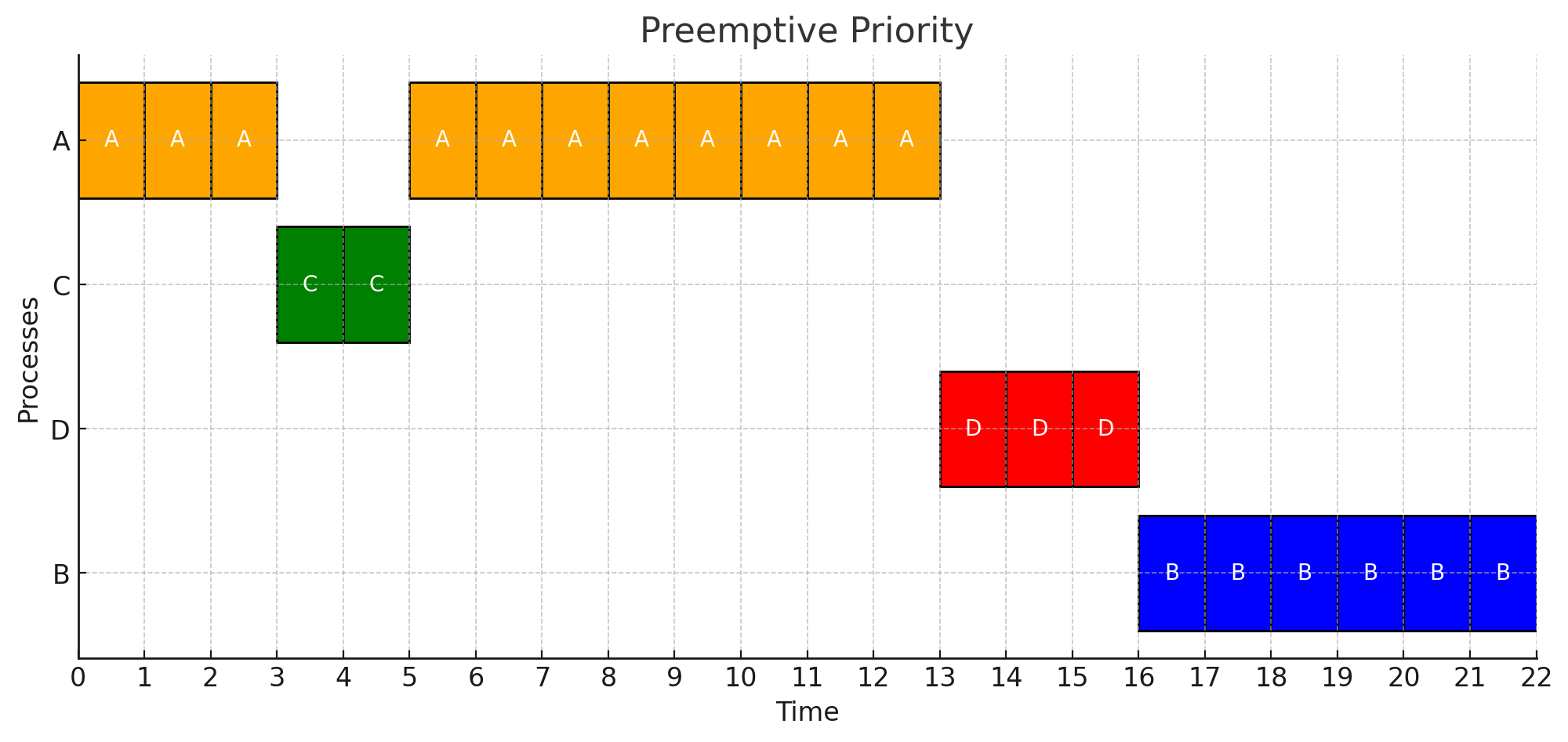
FCFS



Round-Robin



Preemptive Priority-Based



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Waiting Time | A | B | C | D | Total | Average |
| FCFS | 0 | 11 | 14 | 15 | 40 | 10 |
| RR | 10 | 10 | 1 | 6 | 27 | 6.75 |
| PP | 2 | 16 | 0 | 10 | 28 | 7 |

A scheduling algorithm with a lower waiting time is the Shortest Remaining Time First (SRTF). SRTF always selects the process with the shortest remaining burst time to execute next, minimizing the average waiting time.