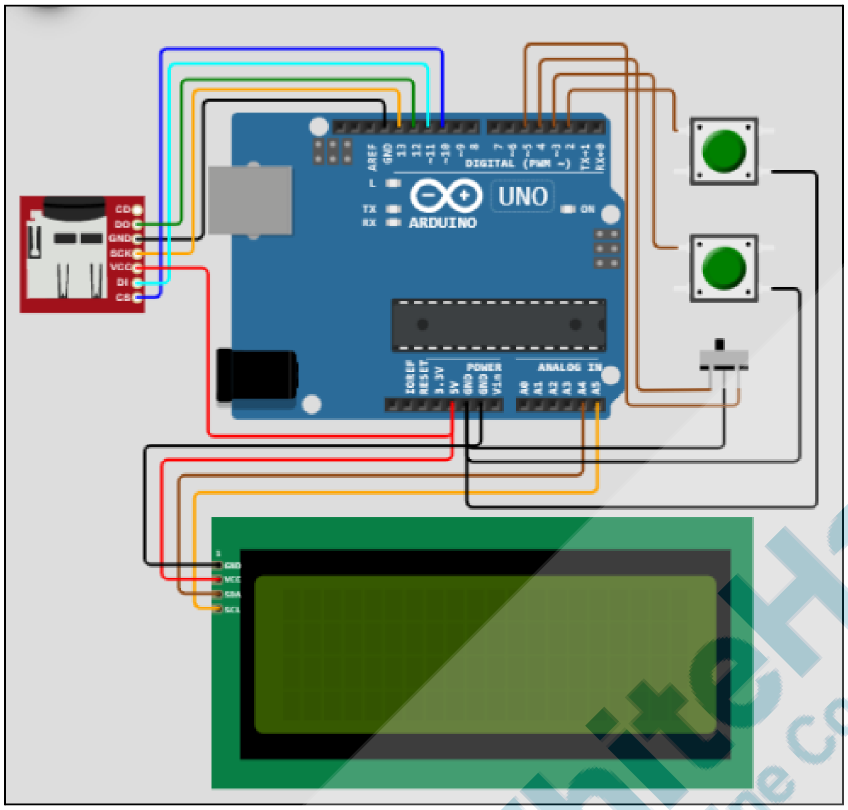
Topic	Digital Book 2	
Class Description	**Students will learn how to generate customized character arrays for LCD displays. Using this knowledge, we will design a welcome animation for our reading device. Additionally, we will display the story page by page on the LCD device.**	
Class	**PRO C274**	
Class time	**50 mins**	
Goal	• Understanding how to generate customized character arrays for LCD display. • Create a welcome animation for the reading device. • Display the stories page-wise on the LCD display.	
Resources Required	• Teacher Resources: ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen ○ Smartphone • Student Resources: ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen	
Class structure	**Warm-Up** **Teacher-Led Activity** **Student-Led Activity** **Wrap-Up**	**10 mins** **15 mins** **15 mins** **10 mins**

WARM-UP SESSION - 10 mins

Teacher Action	Student Action

Hey <student's name>. How are you? It's great to see you! Are you excited to learn something new today? **Following are the WARM-UP session deliverables:** ● Greet the student. ● Revision of previous class activities. ● Quizzes.	**ESR**: Hi, thanks! Yes, I am excited about it! Click on the slide show tab and present the slides
WARM-UP QUIZ Click on In-Class Quiz	
Activity Details **Following are the session deliverables:** ● Appreciate the student. ● Narrate the story by using hand gestures and voice modulation methods to bring in more interest in students.	
TEACHER-LED ACTIVITY 15 mins	
Teacher Initiates Screen Share	
● **Understanding how to generate customized character arrays for LCD display.** ● **Create a welcome animation.**	
Teacher Action	**Student Action**
Do you remember what we did in the last class? Exactly! We learnt that Arduino have 3 kinds of memory. Do you remember what those were? Absolutely correct! But to store a large amount of data, we used an external storage unit. Do you remember which one?	**ESR** : Yes! We learnt about memory. **ESR** : Flash memory, Static RAM, EEPROM

Perfect. It's great that you remember it! Let's get started with today's class. We want to build a reading device with an LCD display using Arduino. Let's look at the boilerplate code first. *Teacher downloads the boilerplate code from [Teacher Activity 2](#).* We have already learned about LCD displays in some of the previous classes. An LCD display is already connected to the Arduino board for our convenience.	**ESR** : We used an SD card.

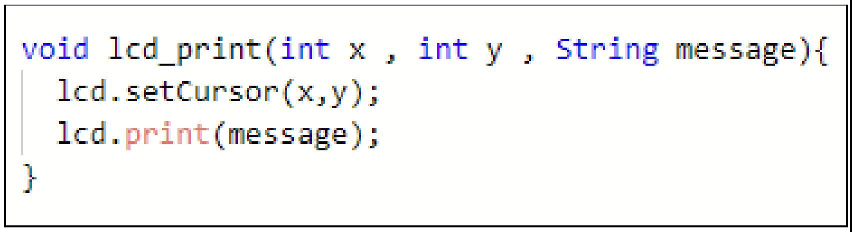


The code provided shows a welcome text on the LCD display.

Teacher goes through the sketch.ino while explaining it.

In the boilerplate code, **lcd_print()** method is already defined. This method takes 3 inputs - 2 integers which gives us the position at which we want to start printing the character and a String which holds the text to print.

Within the method, we call the **setCursor()** method and pass x,y. Then, we call the **print()** method to display the message at that position.



```
void lcd_print(int x , int y , String message){
  lcd.setCursor(x,y);
  lcd.print(message);
}
```

We will use the **lcd_print()** method when we need to print anything on the LCD display.

After that, the **welcome()** method is defined which prints "welcome to digibook" text on screen.

Now, we want to add some welcome animation as well.

To do that, first let's understand each grid in the LCD display.

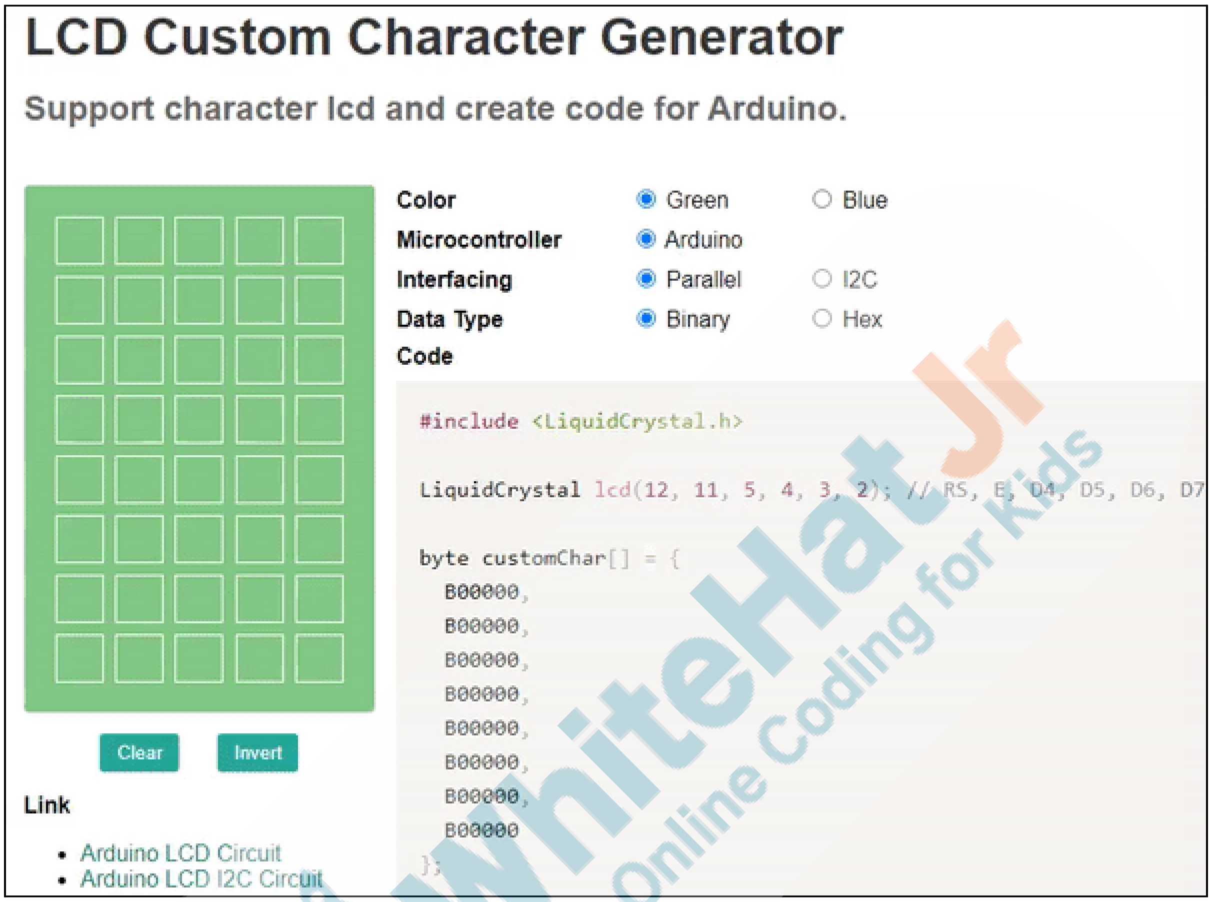
We know that in our LCD there are 20 columns and 4 rows which makes 80 grids in total.

Each of these 80 grids are again made up of 8 rows and 5 columns.

To display character in one of these 80 grids, we manipulate these 8x5 smaller sections which makes each character.

Teacher opens Teacher Activity 3

Here, the grid is broken down into rows and columns. On the right hand side an array is given which helps us to store 8 values- one value for each row.


LCD Custom Character Generator

Support character lcd and create code for Arduino.

Color [✓ Green] Blue
Microcontroller [✓ Arduino]
Interfacing [✓ Parallel] I2C
Data Type [✓ Binary] Hex
Code

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2); // RS, E, D4, D5, D6, D7

byte customChar[] = {
  B00000,
  B00000,
  B00000,
  B00000,
  B00000,
  B00000,
  B00000,
  B00000
};
```

Link

- Arduino LCD Circuit
- Arduino LCD I2C Circuit

Click here to view the reference video.

Click on a grid to make it light up. If we light up any of these characters that we want, **customChar[]** array 8 of these values depending on what character we want to show.

In the boilerplate, we have a file named **lcd_custom_character.h** which includes 5 array animations like- **left_stick**, **right_stick**, **open_eye**, **middle_stick**, **closed_eye** etc. animations.

We will use these arrays in our to display a welcome animation.

Let's start with the code.	

Code:

1. Include the "lcd_custom_character.h" file first so that we can .

```
#include "lcd_custom_character.h"
```

2. An object named lcd is already initialized. Let's define a method named **display_char().** This method will help us display a customized character.
    - Add 3 parameters to this method- 2 integers which gives us the position at which we want to start printing the character and an **id** of the customized character to print.

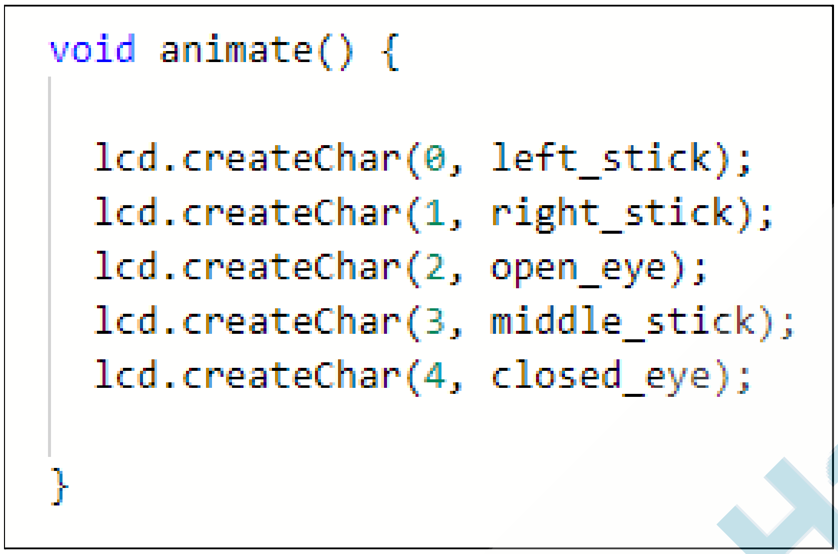
```
void display_char(int x , int y , int id){

}
```

    - Then, we will use the **setCursor()** method to position the cursor and we will use the **write()** method to display the customized character with its id.

```
void display_char(int x , int y , int id){
  lcd.setCursor(x , y);
  lcd.write(id);
}
```

3. Let's define the **animate()** method now. We will use the **createChar()** method to assign an id to the

customized characters.



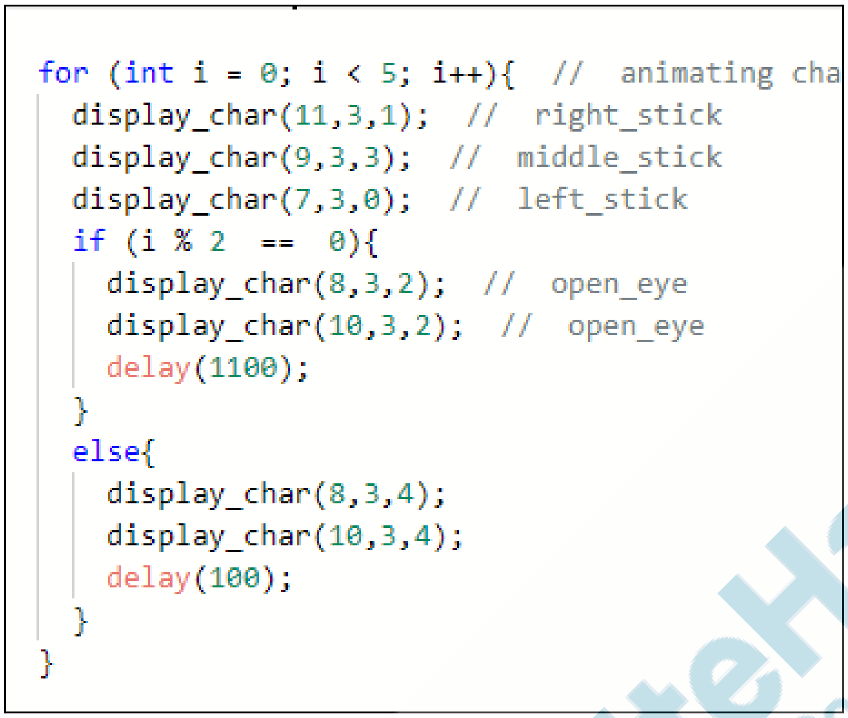
```
void animate() {

  lcd.createChar(0, left_stick);
  lcd.createChar(1, right_stick);
  lcd.createChar(2, open_eye);
  lcd.createChar(3, middle_stick);
  lcd.createChar(4, closed_eye);

}
```

4. Now to display these customized characters, we will use the **display_char()** method.

   Let's say we want to display **left_stick** at the 7th column and 3rd row. We can write,

   ```
   display_char(7,3,0);
   ```

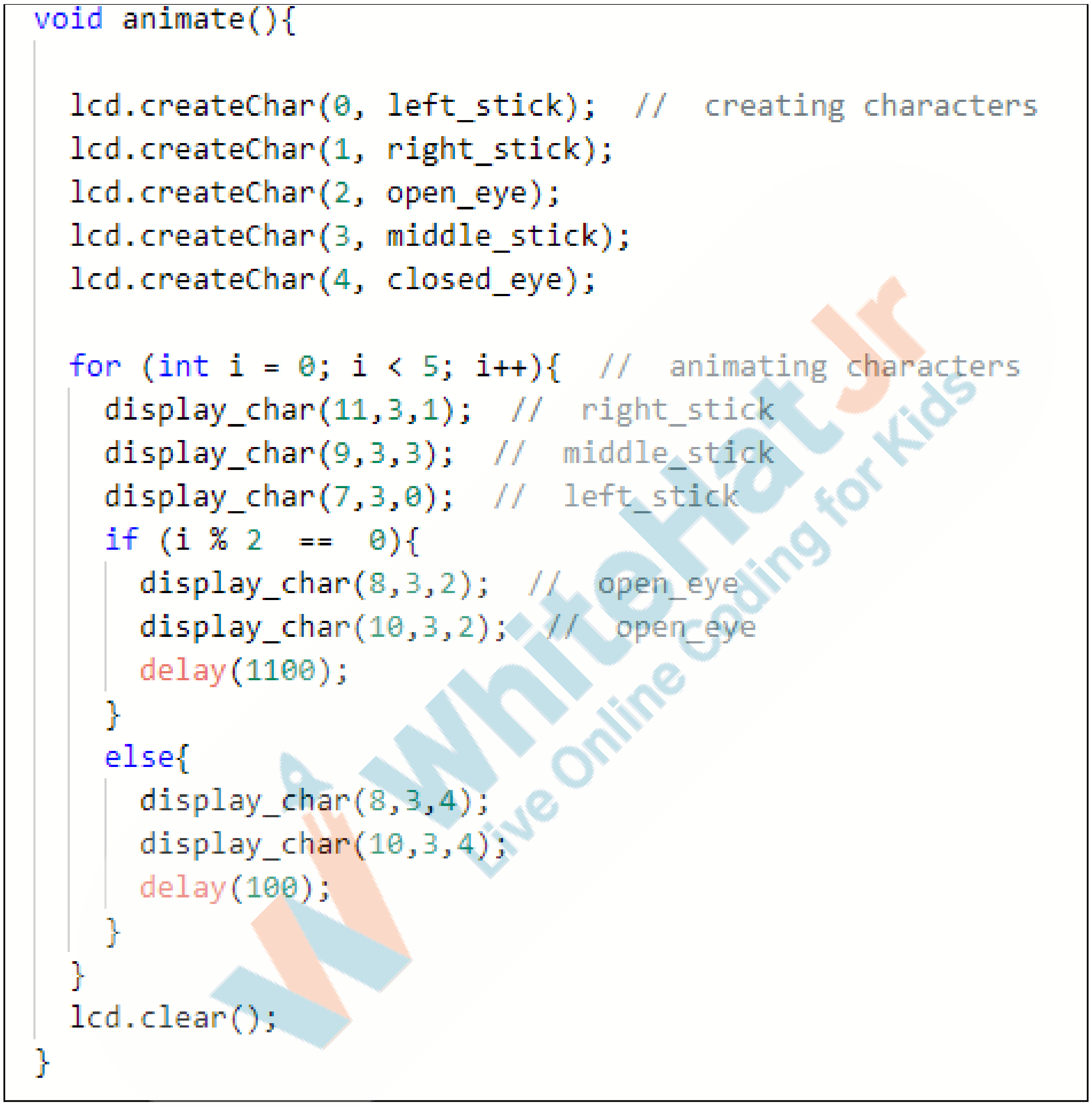
5. We want to create a blinking effect for the eyes of this animation. We have one customized character for open eyes and one for closed eyes. So, we can use a for loop so that we can show closed eyes and open eyes alternatively.



```
for (int i = 0; i < 5; i++){  //  animating cha
  display_char(11,3,1);  //  right_stick
  display_char(9,3,3);  //  middle_stick
  display_char(7,3,0);  //  left_stick
  if (i % 2  ==  0){
    display_char(8,3,2);  //  open_eye
    display_char(10,3,2);  //  open_eye
    delay(1100);
  }
  else{
    display_char(8,3,4);
    display_char(10,3,4);
    delay(100);
  }
}
```

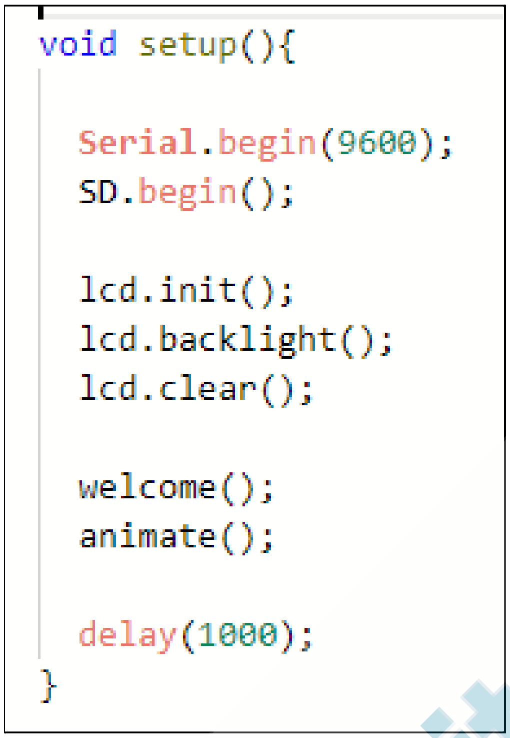
6. Clear the lcd at the end using **lcd.clear()** method.

The **animate()** method should look like this-



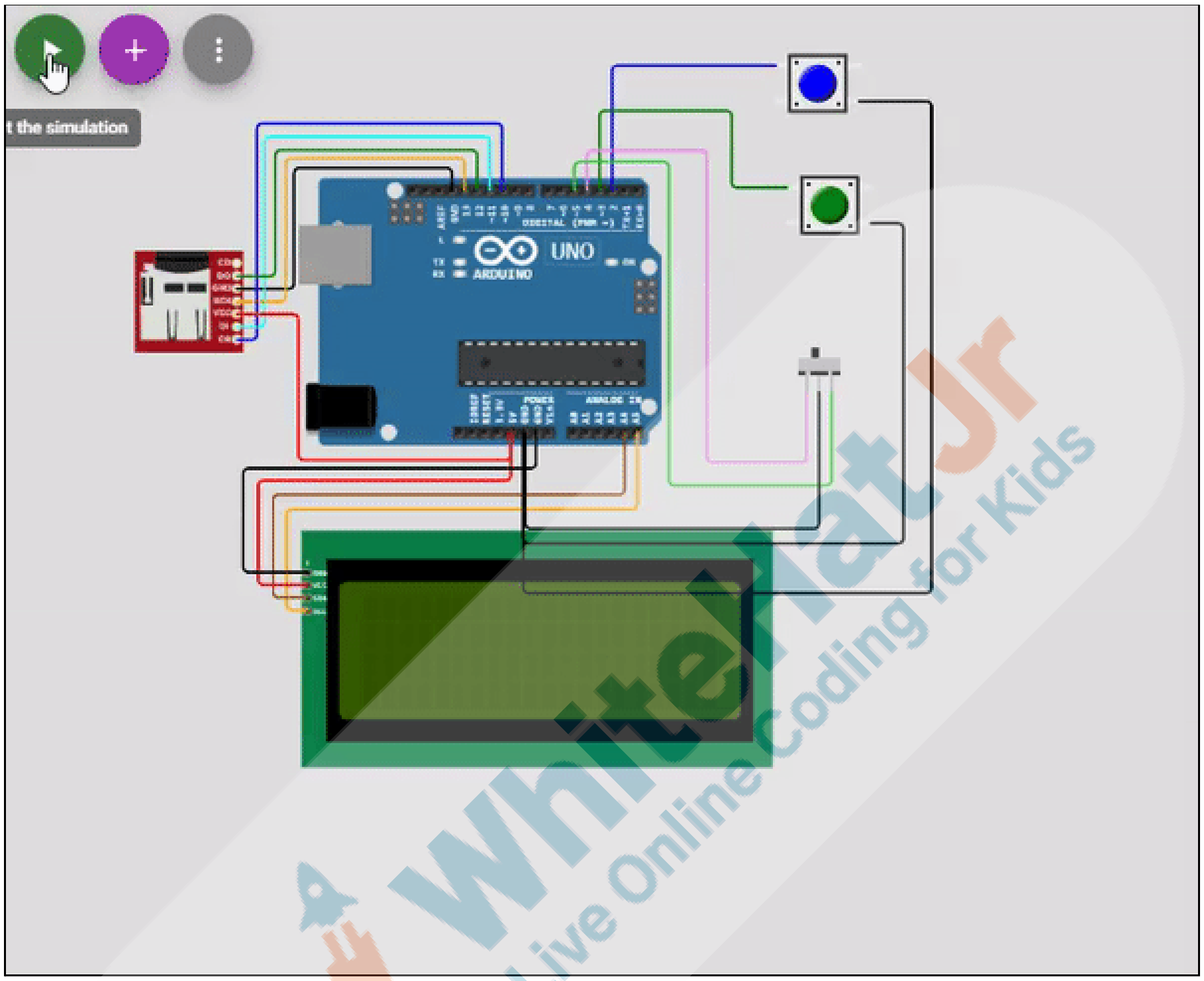
```
void animate(){

  lcd.createChar(0, left_stick);  //  creating characters
  lcd.createChar(1, right_stick);
  lcd.createChar(2, open_eye);
  lcd.createChar(3, middle_stick);
  lcd.createChar(4, closed_eye);

  for (int i = 0; i < 5; i++){  //  animating characters
    display_char(11,3,1);  //  right_stick
    display_char(9,3,3);  //  middle_stick
    display_char(7,3,0);  //  left_stick
    if (i % 2  ==  0){
      display_char(8,3,2);  //  open_eye
      display_char(10,3,2);  //  open_eye
      delay(1100);
    }
    else{
      display_char(8,3,4);
      display_char(10,3,4);
      delay(100);
    }
  }
  lcd.clear();
}
```

Let’s call the **animate()** method in the **setup()** method.	



```
void setup(){

  Serial.begin(9600);
  SD.begin();

  lcd.init();
  lcd.backlight();
  lcd.clear();

  welcome();
  animate();

  delay(1000);
}
```

Reference output:



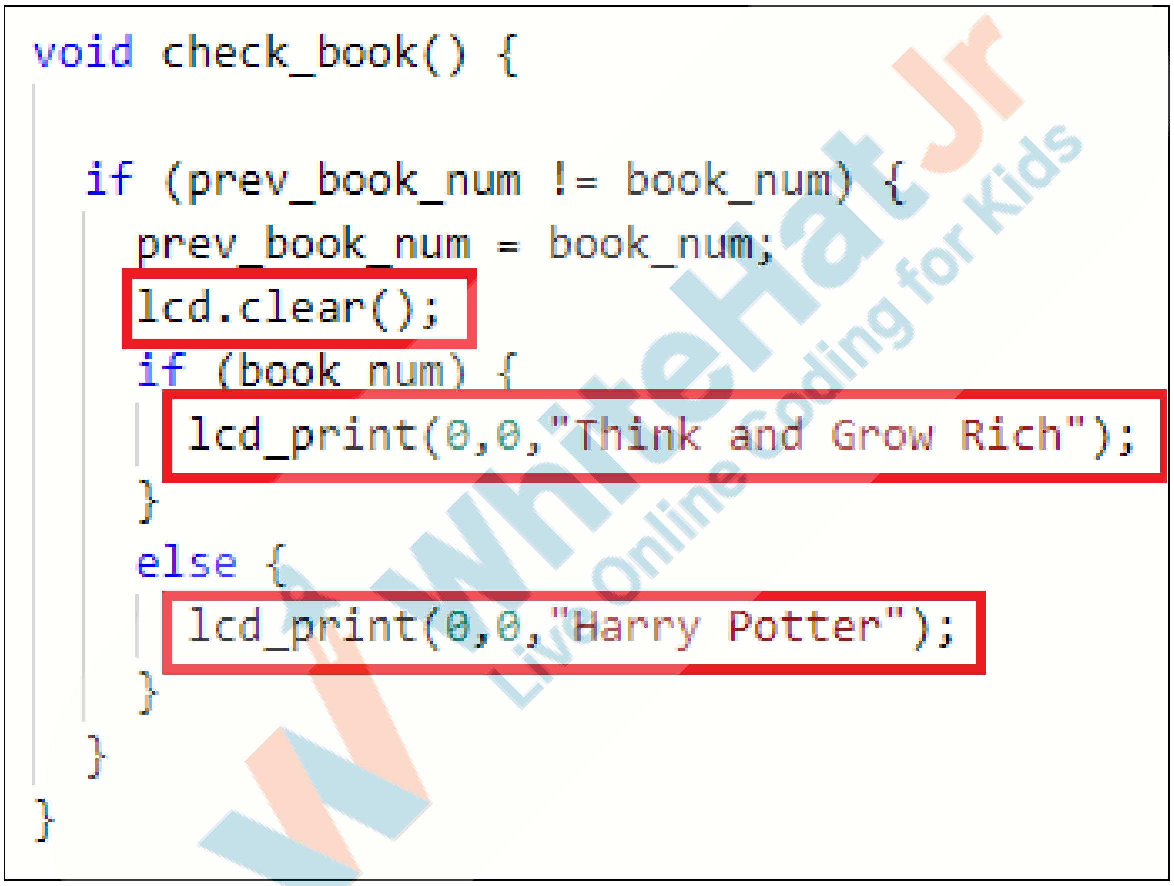
Click here to view the reference video.

Now, we have another challenge.We were showing the book data on the serial monitor in our previous class. Today we want to show the data on the LCD display screen. Can you write the code for it? Let's try. I will guide you through it.	

Teacher Stops Screen Share	
STUDENT-LED ACTIVITY- 15 mins	
• **Ask the student to press the ESC key to come back to the panel.** • **Guide the student to start Screen Share.** • **The teacher gets into Full Screen.**	
Student Initiates Screen Share	
ACTIVITY • **Change the code to display the story page-wise** • **Display the story on the LCD display**	
Teacher Action	Student Action
Open the wokwi simulator link and, • Create a new Arduino project. • Delete all the previously existing files **(sketch.ino, diagram.json)** from your project. • Open the Student Activity 2 and download all the files from it. • Upload all the downloaded files that you have just downloaded in the previous step into your project. • Run the simulation once.	
Right now, the book name is being shown on the Serial monitor. But we want to show it on the LCD display. What should we do? Did we already define any method which helps us print	**ESR:** This portion of code was written in the **check_book()** method. We need to replace the **Serial.print()** methods there.

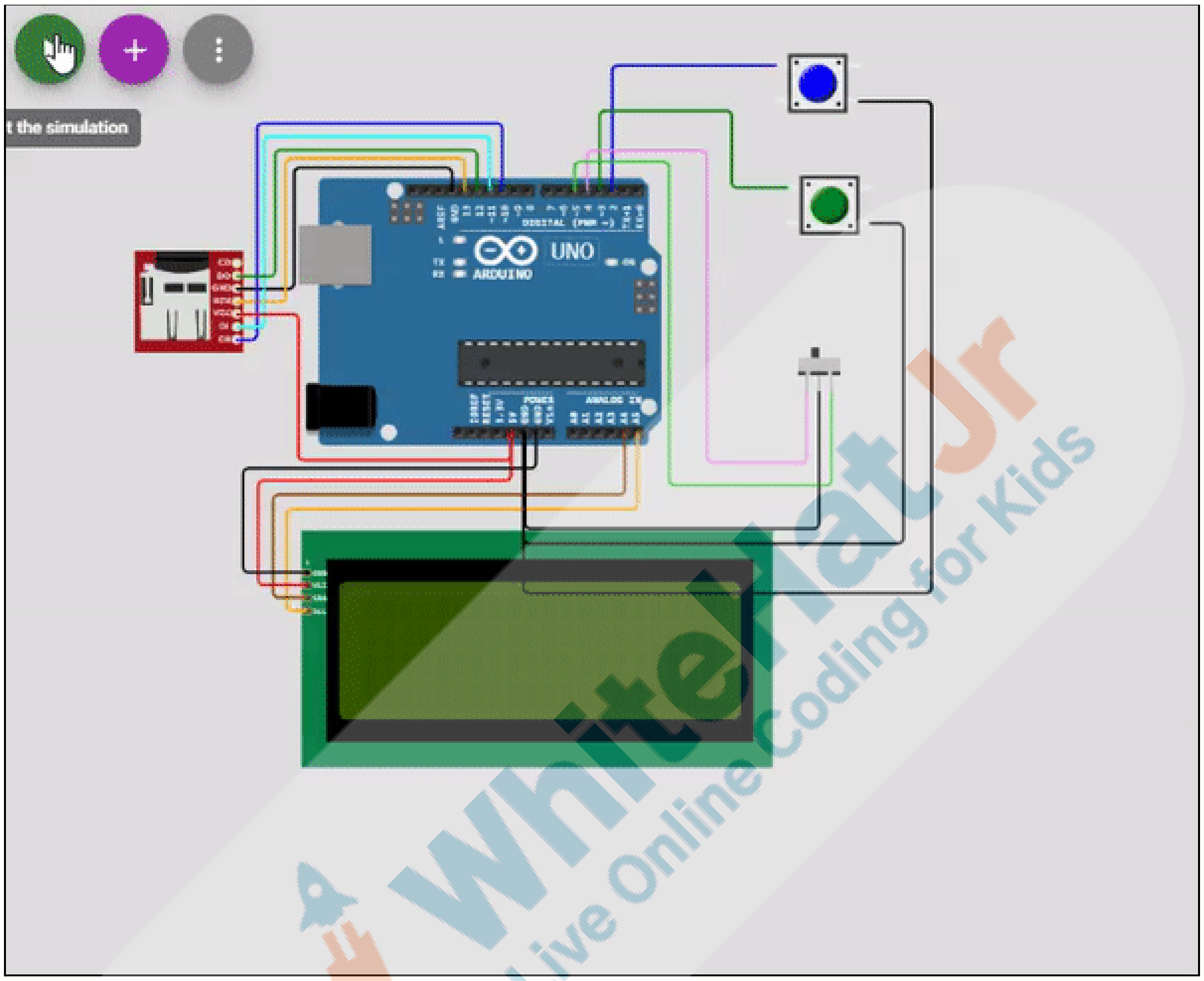
characters on the LCD display? Great! Go ahead and write the code.	**ESR:** We have the **lcd_print()** method. We can use it.

Reference code:



```
void check_book() {

  if (prev_book_num != book_num) {
    prev_book_num = book_num;
    lcd.clear();
    if (book_num) {
      lcd_print(0,0,"Think and Grow Rich");
    }
    else {
      lcd_print(0,0,"Harry Potter");
    }
  }
}
```

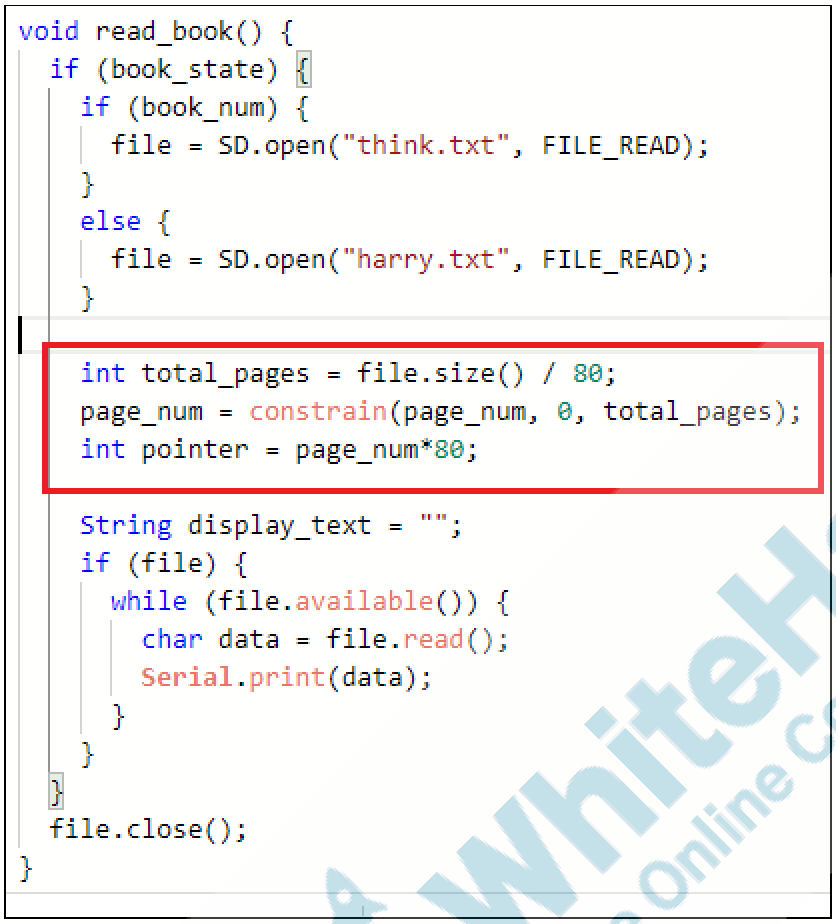
Reference output:



Click here to view the reference video.

Do you see any other problem? Exactly. The user can’t even read it. How can we fix this problem?	**ESR:** Yes. The story is being printed on the serial monitor. Also, it’s too fast to read. **ESR:** We can break it down into pages according to our

Perfect! The LCD display can print 80 characters at a time. So, we will break down the book into 80 characters per page.
 	
Let's rewrite the **read_book()** method.
 	
1. First, initiate 2 variables - **page_num** and **prev_page_num**. This is to be initiated as a global variable.
`int page_num = 0 , prev_page_num = -1;`
 	
2. In the **read_book()** method, let's write code to calculate the total number of pages in a book.
`int total_pages = file.size() / 80;`
 	
3. The **page_num** variable couldn't go beyond **total_pages.** So, add a constraint.
`page_num = constrain(page_num, 0, total_pages);`
 	
4. We will also need a variable which holds the character in the book from where we want print when we are scrolling through the book. This character should be the first character of the page we are on.
`int pointer = page_num*80;`
	LCD display's capacity.



```
void read_book() {
  if (book_state) {
    if (book_num) {
      file = SD.open("think.txt", FILE_READ);
    }
    else {
      file = SD.open("harry.txt", FILE_READ);
    }

    int total_pages = file.size() / 80;
    page_num = constrain(page_num, 0, total_pages);
    int pointer = page_num*80;

    String display_text = "";
    if (file) {
      while (file.available()) {
        char data = file.read();
        Serial.print(data);
      }
    }
  }
  file.close();
}
```

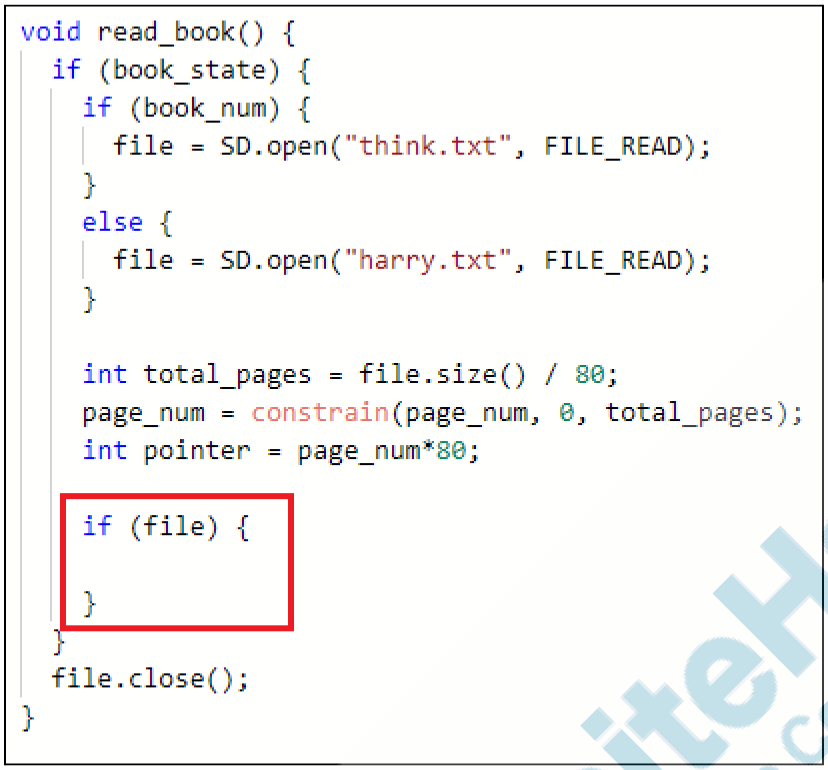
5. The **display_text** variable holds the data to display. We want to use it in one more method today, so we need to make this variable global.

   So, move

   ```
   String display_text = "";
   ```

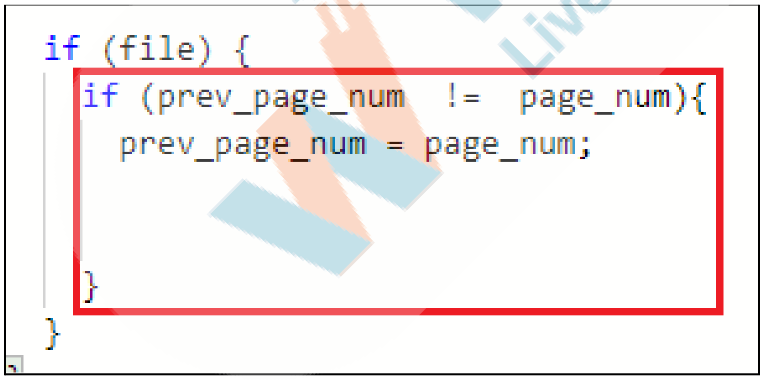
   to the top.

6. Let's get rid of the **while()** loop inside the **if** condition for now.



```
void read_book() {
  if (book_state) {
    if (book_num) {
      file = SD.open("think.txt", FILE_READ);
    }
    else {
      file = SD.open("harry.txt", FILE_READ);
    }

    int total_pages = file.size() / 80;
    page_num = constrain(page_num, 0, total_pages);
    int pointer = page_num*80;

    if (file) {

    }
  }
  file.close();
}
```

7. Inside this if condition, we need to keep track of the pages. This portion of code is similar to what we had done in the **check_book()** method.



```
if (file) {
  if (prev_page_num  !=  page_num){
    prev_page_num = page_num;


  }
}
```

8. Here, we will clear up **display_text**. It might hold data from the previous page.

```
display_text = "";
```

9. Now, go to the first character of the current page. We can use the **seek()** method for that. **seek()** method helps us to seek a new position in the file.

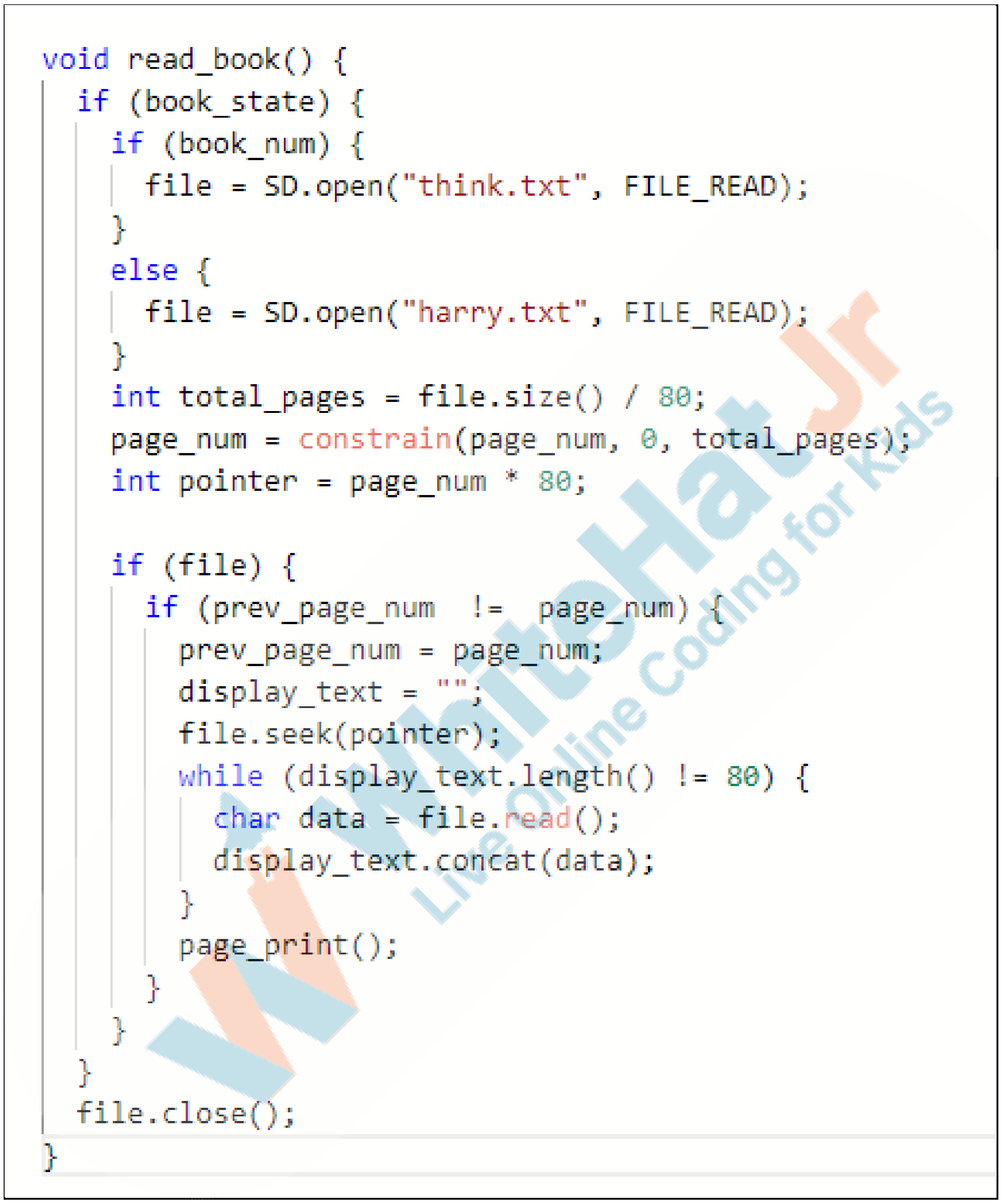
```
file.seek(pointer);
```

10. Now, we are at the beginning of our page. We want to display the next 80 characters. For that, let's store next 80 characters in the **display_text** variable.

```
 while (display_text.length() != 80){

        char data = file.read();

        display_text.concat(data);

  }
```

11. Now, we need to print this **display_text.** To do that let's call a method called **page_print().** We will define this method in the next step.

This is how the **read_book()** method should look like at this point-



```
void read_book() {
  if (book_state) {
    if (book_num) {
      file = SD.open("think.txt", FILE_READ);
    }
    else {
      file = SD.open("harry.txt", FILE_READ);
    }
    int total_pages = file.size() / 80;
    page_num = constrain(page_num, 0, total_pages);
    int pointer = page_num * 80;

    if (file) {
      if (prev_page_num  !=  page_num) {
        prev_page_num = page_num;
        display_text = "";
        file.seek(pointer);
        while (display_text.length() != 80) {
          char data = file.read();
          display_text.concat(data);
        }
        page_print();
      }
    }
  }
  file.close();
}
```

12. Let's define the **page_print()** method.
    Our LCD display has 20 columns and 4 rows. We

have 80 characters to show. So, we will print 20 characters in each of these 4 rows. We will use a **for** loop to do this.

The **page_print()** method should look like this-

```
void page_print(){
  for (int i = 0; i < 4; i++){
    lcd_print(0,i,display_text.substring(i*20 ,(i+1)*20));
  }
}
```

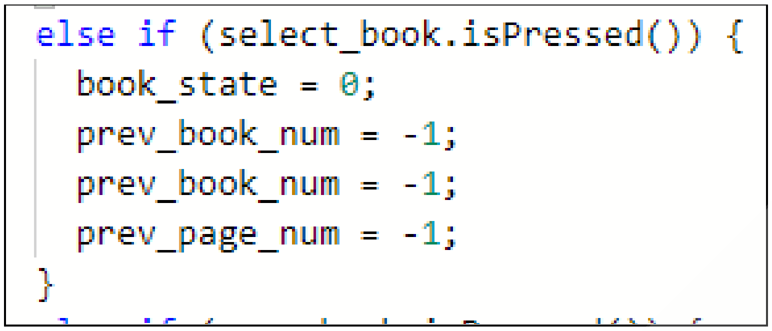
13. To change the **page_num**, we will use the **next** and **prev** buttons. But these two buttons were already programmed to change books.

    So, we will write the code in a way that these two buttons change books when book_state is 0. When the book_state is 1, the **next** and **prev** buttons should change the pages.

    This should be written in the **loop()** method.

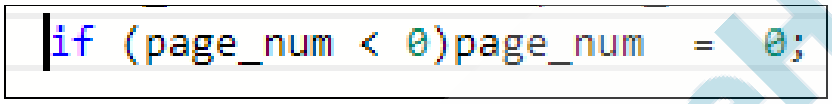
```
if (next.isPressed()){
  if (book_state)page_num++;
  else book_num++;
}
else if (prev.isPressed()){
  if (book_state)page_num--;
  else book_num--;
}
```

14. Reset the **page_num** and **prev_page_num** variables when the **select_book** button is pressed.



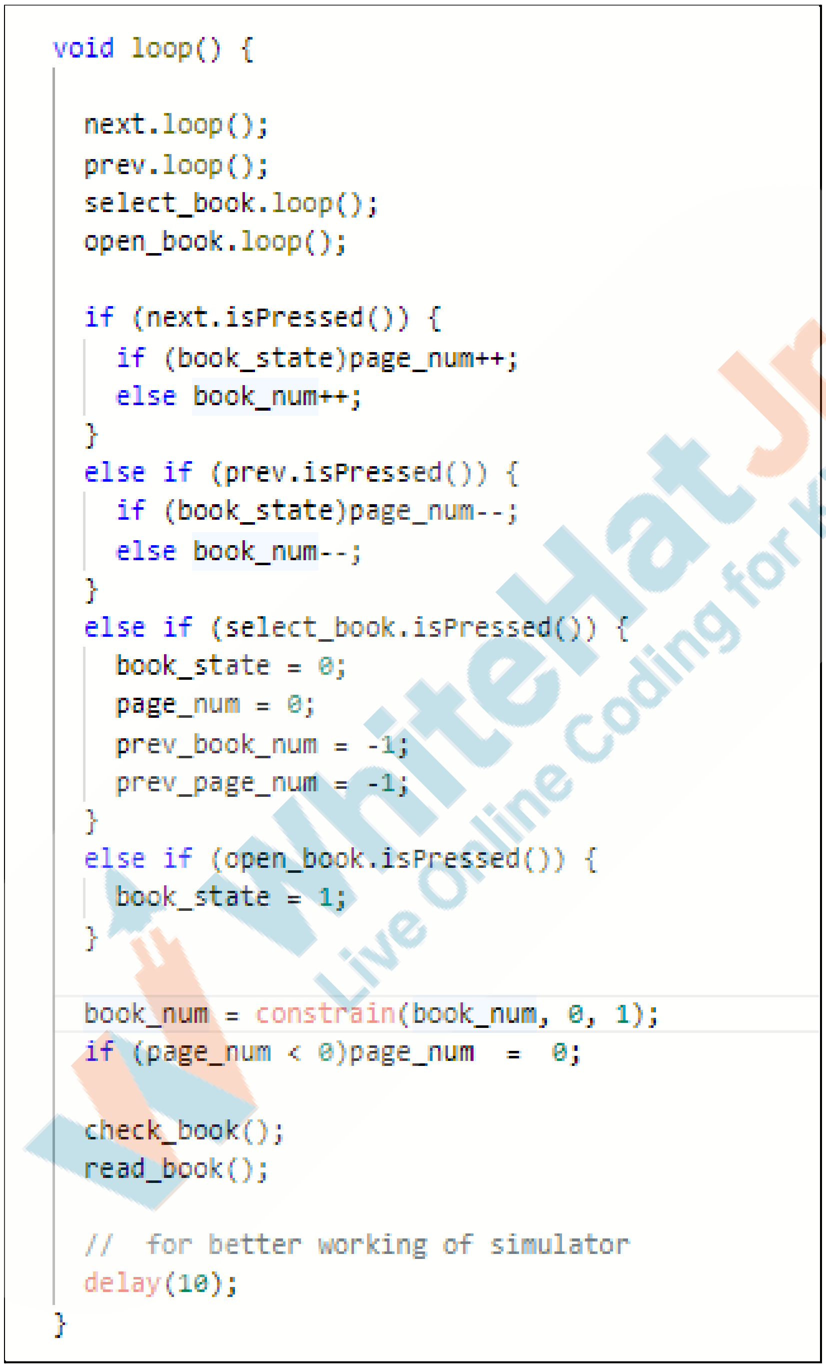
```
else if (select_book.isPressed()) {
  book_state = 0;
  prev_book_num = -1;
  prev_book_num = -1;
  prev_page_num = -1;
}
```

15. We will also make sure that the **page_num** variable doesn't go below 0.



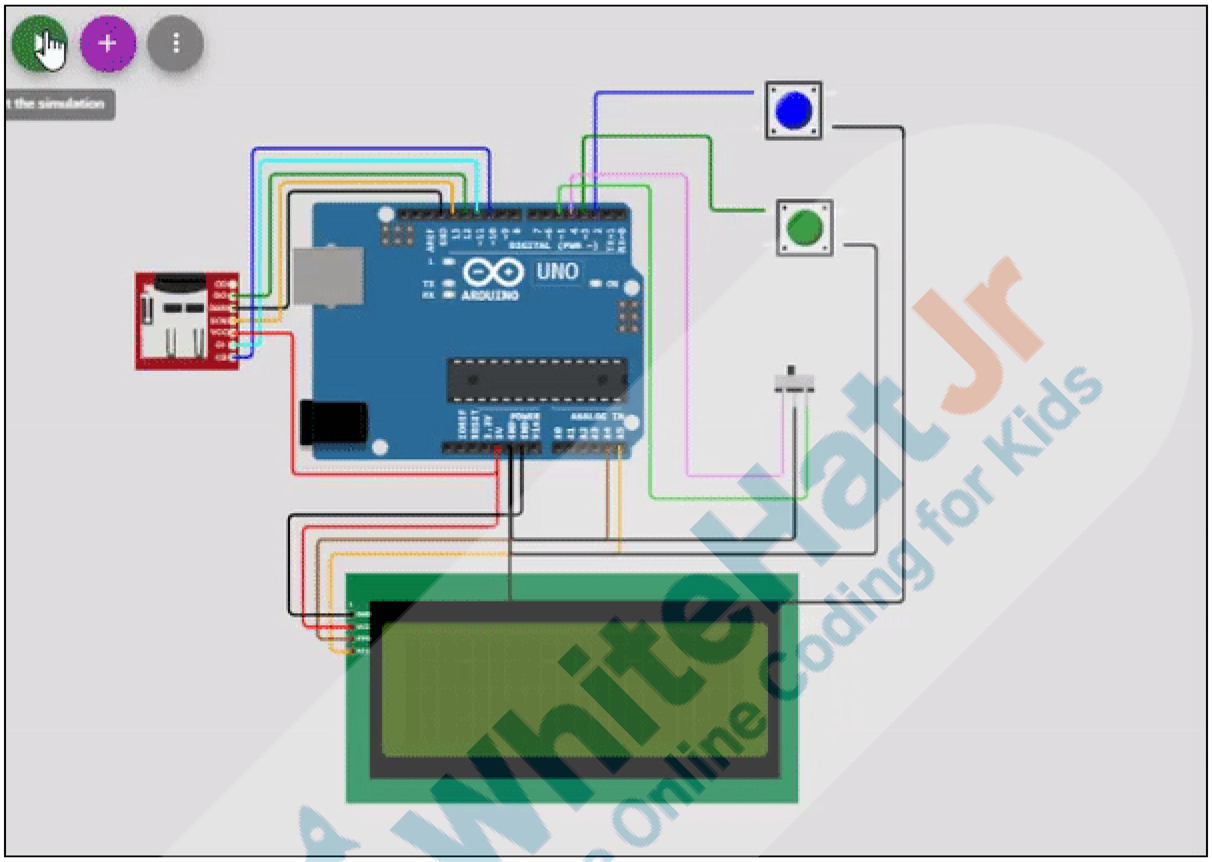
```
if (page_num < 0)page_num  =  0;
```

The **loop()** method should look like this-



```
void loop() {

  next.loop();
  prev.loop();
  select_book.loop();
  open_book.loop();

  if (next.isPressed()) {
    if (book_state)page_num++;
    else book_num++;
  }
  else if (prev.isPressed()) {
    if (book_state)page_num--;
    else book_num--;
  }
  else if (select_book.isPressed()) {
    book_state = 0;
    page_num = 0;
    prev_book_num = -1;
    prev_page_num = -1;
  }
  else if (open_book.isPressed()) {
    book_state = 1;
  }

  book_num = constrain(book_num, 0, 1);
  if (page_num < 0)page_num  =  0;

  check_book();
  read_book();

  //  for better working of simulator
  delay(10);
}
```

Reference output:



Click here to view the reference video.

Teacher Guides Student to Stop Screen Share

WRAP-UP SESSION - 10 mins

Activity details

Following are the WRAP-UP session deliverables:

- Appreciate the student.
- Revise the current class activities.
- Discuss the quizzes.

WRAP-UP QUIZ
Click on In-Class Quiz

Activity Details

Following are the session deliverables:
- Explain the facts and trivia
- Next class challenge
- Project for the day
- Additional Activity (Optional)

FEEDBACK
- **Appreciate and compliment the student for trying to learn a difficult concept.**
- **Get to know how they are feeling after the session.**
- **Review and check their understanding.**

Teacher Action	Student Action
You get "hats-off" for your excellent work! In the next class, we will learn how to use an RTC module and create a smart clock using it.	*Make sure you have given at least 2 hats-off during the class for:*

PROJECT OVERVIEW DISCUSSION
Refer the document below in Activity Links Sections

Teacher Clicks ✖ End Class

ADDITIONAL ACTIVITIES (Optional)	
Additional Activities	

ACTIVITY LINKS		
Activity Name	**Description**	**Links**
Teacher Activity 1	Simulator	wokwi
Teacher Activity 2	Teacher Boilerplate code	https://github.com/procodingclass/PRO-C274-Teacher-Boilerplate
Teacher Activity 3	Generating customized character array for LCD	https://maxpromer.github.io/LCD-Character-Creator/
Teacher Reference 1	Teacher Reference Code	https://github.com/procodingclass/PRO-274-Reference-Code
Student Activity 1	Simulator	wokwi
Student Activity 2	Boilerplate link	https://github.com/procodingclass/PRO-C274-Student-Boilerplate
Teacher Reference 3	Project	https://s3-whjr-curriculum-uploads.whjr.online/ee221618-17d9-4670-9c7e-43dc7756ae3f.pdf
Teacher Reference 4	Project Solution	https://github.com/procodingclass/PRO-C274-Project-Solution
Teacher Reference 5	In-Class Quiz	https://s3-whjr-curriculum-uploads.whjr.online/26f9c65b-8ed9-4b37-8cdf-ee54bf643d4c.pdf