Topic	**KEYPAD**	
Class Description	**Students will learn about 4X4 Matrix KEYPAD and develop password application on Keypad**	
Class	**PRO C259**	
Class time	**50 mins**	
Goal	● Keypad Connections ● Keypad Working ● Secret Password	
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**
Credit & Permissions:	**Code samples used for Firebase-Google Authentication are licensed under the Apache 2.0 License.** **Expo documentation used from - https://expo.io** **Note: Keep this row section only if applicable**	
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 15mins	
Student Initiates Screen Share	
● **Algorithm to calculate vote count**	
Teacher Action	**Student Action**
So, in the last class, we designed our Electronic Voting Machine. Do you have any doubts? *If the student has any doubts, clarify the doubts* Do you use the phone?	ESR: Varied!

How do you use the application or call someone? If you want to type something you need to use a touchpad and it will open the default keypad. Am I right? Not only phones, but other devices like calculators, microwaves, ovens, and door locks. They are practically everywhere. So have you ever think how this keypad works? So you want to learn how this keypad actually works? Let's learn what will happen behind the scenes when we pressed the button and how it displays on the screen. So today we are using Membrane keypads which are made of a thin, flexible membrane material. They do come in many sizes 4×3, 4×4, 4×1, etc. Regardless of their size, they all work in the same way	**ESR**: Varied! **ESR**: Yes! **ESR**: Varied!

1 2 3 A 4 5 6 B 7 8 9 C * 0 # D	
Teachers click on Teacher Activity 1	Student clicks on Student Activity 1
Write the keypad in the search bar under the component section and show it to the student.	
Teachers click on Teacher Activity 2	Student clicks on Student Activity 2
Now we will see how this keypad works actually: Let's take an example of a 4 x 4 keypad. Here 4x4 means 4 rows (R1 - R4) and 4 columns(C1-C4). In total it has 16 keys. Beneath each key, there is a special membrane switch.	

All these membrane switches are connected to each other with conductive trace underneath the pad forming a matrix of a 4×4 grid. The working principle is very simple. Pressing a button shorts one of the row lines to one of the column lines, allowing current to flow between them. A microcontroller can scan these lines for a button-pressed state. To do this, it follows the below procedure. 1. Microcontroller sets all the column and row lines to input. 2. Then, it picks a row and sets it HIGH. 3. After that, it checks the column lines one at a time. 4. If the column connection stays LOW, the button on the row has not been pressed. 5. If it goes HIGH, the microcontroller knows which row was set HIGH, and which column was detected HIGH when checked. 6. Finally, it knows which button was pressed that corresponds to the detected row & column.	
Step -1:Select the components ● 1 x ESP32 ● 1 x Keypad: 4 Rows (R1-R4) and 4 Columns (C1-C4)	

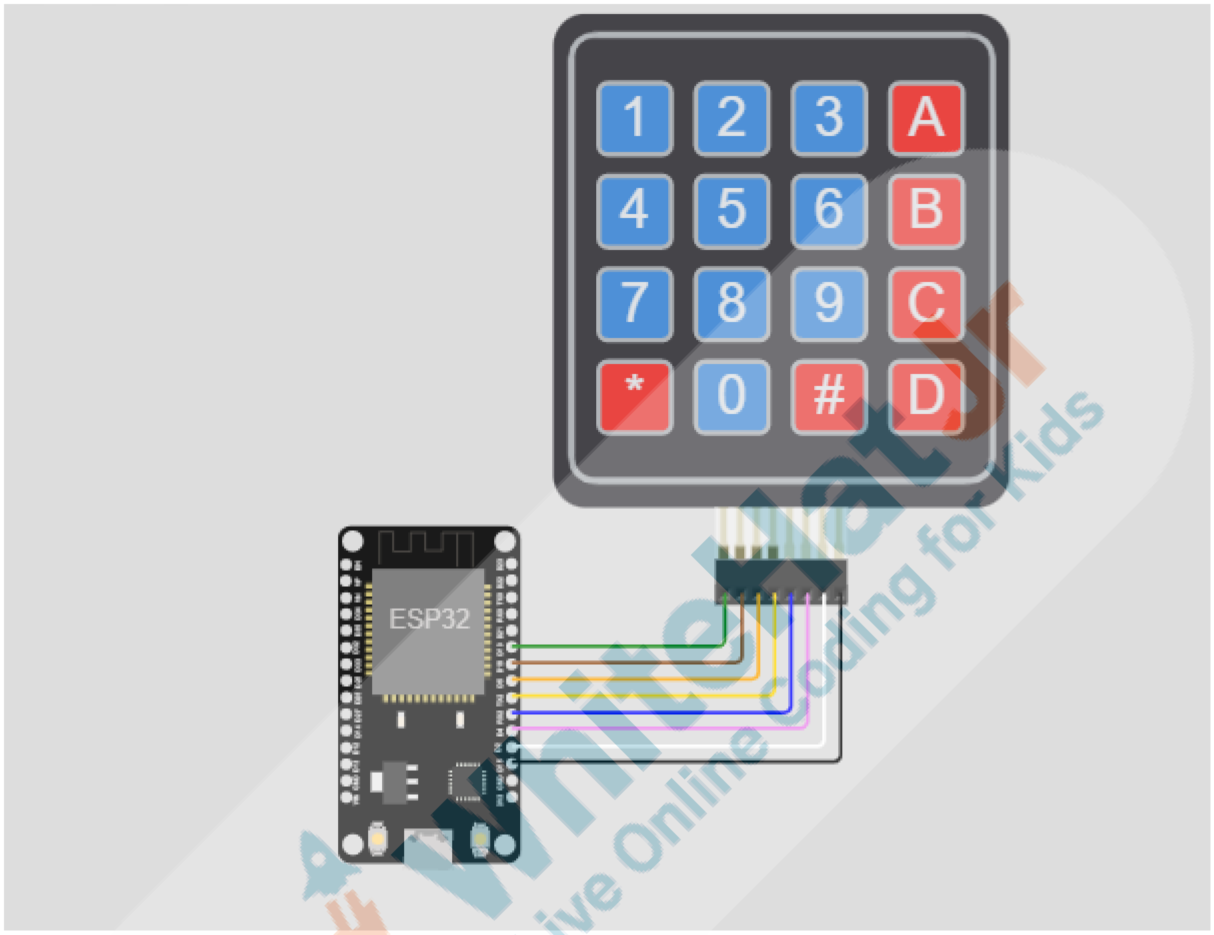
Step -2: Let's do connections:

The circuit of this project consists of an ESP32 Controller, & a Keypad.

- Select Keypad from the simulator list. Connect as per the below:

KEYPAD	ESP32 PIN
R1	GPIO 19
R2	GPIO 18
R3	GPIO 5
R4	TX2
C1	RX2
C2	GPIO 4
C3	GPIO 2
C4	GPIO 15

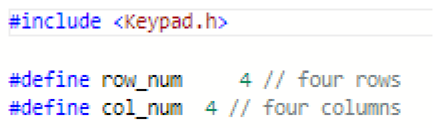
*Note: Wire color can be changed via **diagram.json** file. Go to the diagram.json wire and changed the color of the wire. Any design changes or color changes can be done via the diagram.json file. Keep the track of the component and change the design settings.*



So our next task is to write the Code

Include keypad library to access the keypad application. This library takes care of setting up the pins and pulling the different columns and rows.

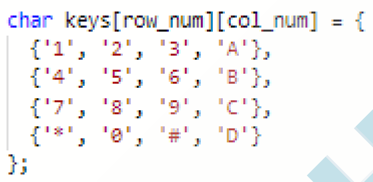
1. set the number of rows and columns on the keypad
2. **define row_num** 4
3. **define col_num** 4
4. As we are using **4 x 4** matrix



```
#include <Keypad.h>

#define row_num     4 // four rows
#define col_num  4 // four columns
```

char is a keyword, **keys** is a variable, write all the keys map array for both Rows and Columns.

Note: Write in the same way as written on the keypad

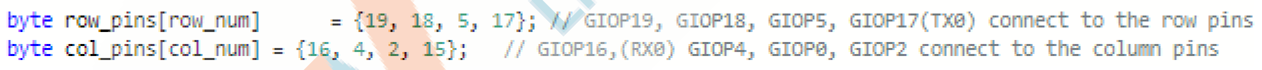


```
char keys[row_num][col_num] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};
```

Define all the pin numbers for row and column

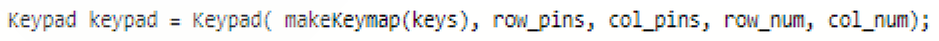
1. Byte is the keyword that is used to store **row_pins, row_pins** is the variable which will store all the rows pins.

2. **col_pins(col_num)** which will store all the column pins.



```
byte row_pins[row_num]      = {19, 18, 5, 17}; // GIOP19, GIOP18, GIOP5, GIOP17(TX0) connect to the row pins
byte col_pins[col_num] = {16, 4, 2, 15};   // GIOP16,(RX0) GIOP4, GIOP0, GIOP2 connect to the column pins
```

Create the keypad object for the keypad class to access all the keys. It will access all **rows_pins,col_pins** along with **row_num** and **col_num**



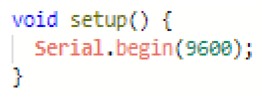
```
Keypad keypad = Keypad( makeKeymap(keys), row_pins, col_pins, row_num, col_num);
```

Initialize using **void setup()** function

1. **void setup()** is used to initialize

2. **Serial.begin() Serial.begin(9600)** is used for data exchange data speed. This tells the Arduino to get

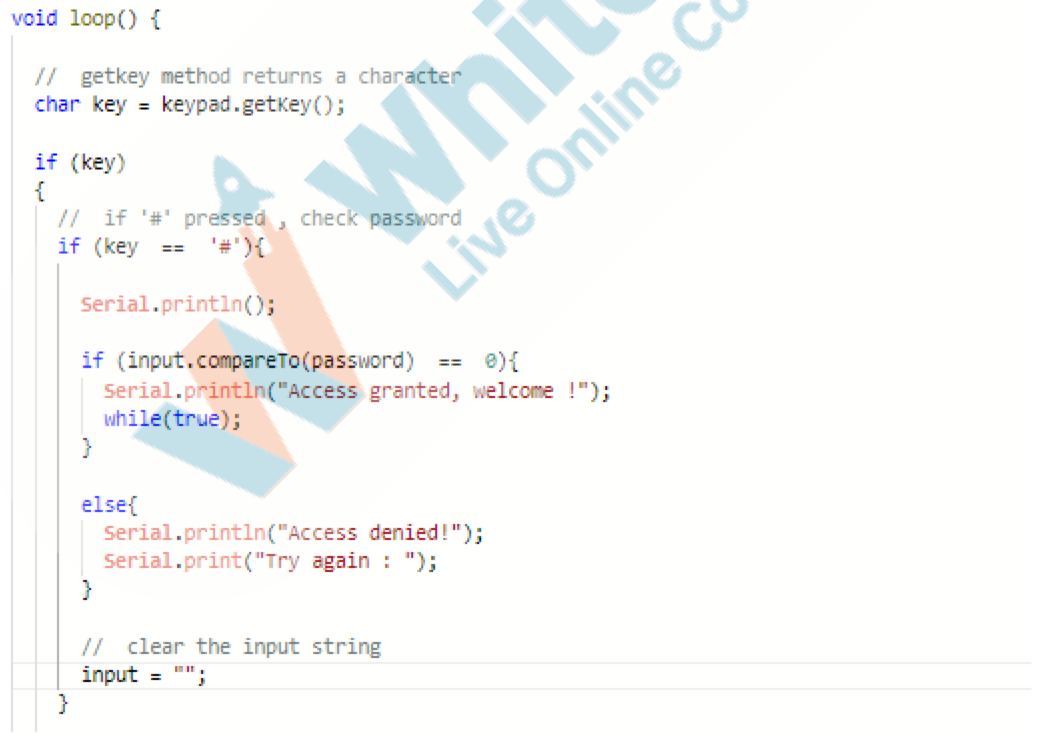
ready to exchange messages with the Serial Monitor at a data rate of 9600 bits per second. That's 9600 binary ones or zeros per second and is commonly called a baud rate.



```
void setup() {
  Serial.begin(9600);
}
```

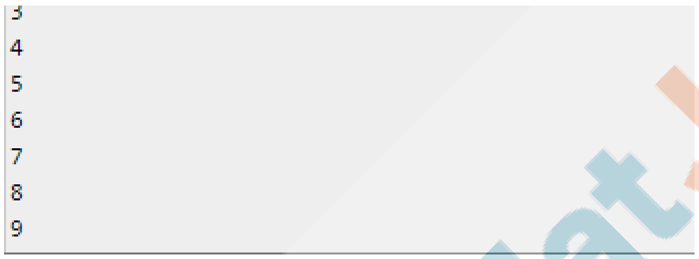
Execution of the main process:

- **void loop()** function is used to execute the main process.
- **Keypad.getKey()** returns the keycode of the pressed key,. If the key is pressed return the keycode. The keycode is retrieved from the keymap array.



```
void loop() {

  //  getkey method returns a character
  char key = keypad.getKey();

  if (key)
  {
    //  if '#' pressed , check password
    if (key  ==  '#'){

      Serial.println();

      if (input.compareTo(password)  ==  0){
        Serial.println("Access granted, welcome !");
        while(true);
      }

      else{
        Serial.println("Access denied!");
        Serial.print("Try again : ");
      }

      //  clear the input string
      input = "";
    }
```

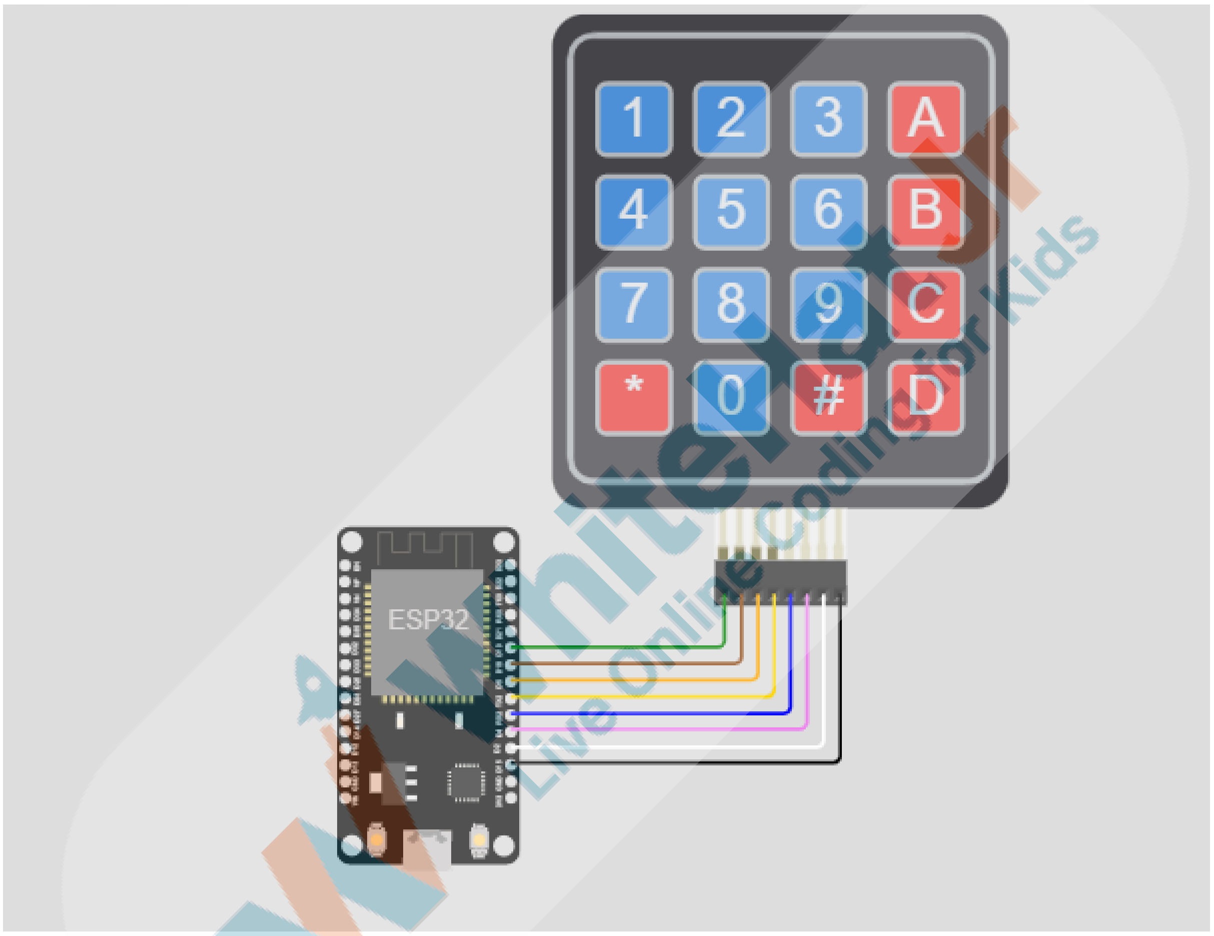
Output:

Click on the Save button and then click on the simulation button 1. Press the key and see the output on the Serial Monitor of the simulator. 2. Just press the keys and you will get the output.	
 3 4 5 6 7 8 9	
Now it's your turn to try this keypad. But do you think trying only will enough this? Let's make an application on the same.	
Student Stops Screen Share	
We have one more class challenge for you. Can you solve it? Let's try. I will guide you through it.	
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	
<u>ACTIVITY</u> • Keypad Connections • Secret Password application	

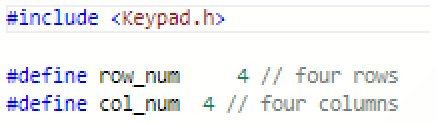
WhiteHat Jr
Live online Coding for Kids

Teacher Action	Student Action
Teacher helps the students	Student click on Student Activity 1
Step -1:Select the components • 1 x ESP32 • 1 x Keypad: 4 Rows (R1-R4) and 4 Columns (C1-C4)	
Step -2: Let's do connections: The circuit of this project consists of an ESP32 Controller, & a Keypad. • Select Keypad from the simulator list. Connect as per the below: (see pin table below) *Note: Wire color can be changed via **diagram.json** file. Go*	

KEYPAD	ESP32 PIN
R1	GPIO 19
R2	GPIO 18
R3	GPIO 5
R4	TX2
C1	RX2
C2	GPIO 4
C3	GPIO 2
C4	GPIO 15

to the diagram.json wire and chang the color of the wire. Any design changes or color changes can be done via the diagram.json file. Keep the track of the component and change the design settings.	
	
So our next task is to write Code 1. Include a keypad library to access the keypad application. This library takes care of setting up the pins and pulling the different columns and rows. 2. set the number of rows and columns on the keypad 3. **define row_num** 4	

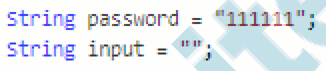
4. **define col_num** 4
5. As we are using **4 x 4** matrix



```
#include <Keypad.h>

#define row_num     4 // four rows
#define col_num  4 // four columns
```

As we need to make a security password lock application, Let's set the password for the same

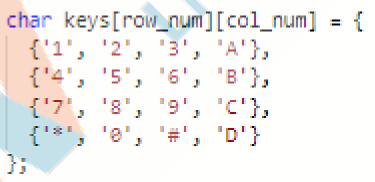
1. **Store password** in variable password i.e "11111", it can be anything. Just save in the string password.
2. **String input** to save the user input from the user and it will match the input with a password.



```
String password = "111111";
String input = "";
```

char is a keyword, **keys** is a variable, write all the keys map array for both Rows and Columns.

Note: Write in the same way as written on the keypad



```
char keys[row_num][col_num] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};
```

Define all the pin numbers for row and column

1. Byte is the keyword that is used to store **row_pins, row_pins** is the variable that will store all the rows pins.
2. **col_pins(col_num)** which will store all the column pins.

```
byte row_pins[row_num]      = {19, 18, 5, 17}; // GIOP19, GIOP18, GIOP5, GIOP17(TX0) connect to the row pins
byte col_pins[col_num] = {16, 4, 2, 15};   // GIOP16,(RX0) GIOP4, GIOP0, GIOP2 connect to the column pins
```

Create the keypad object for the keypad class to access all the keys.It will access all **rows_pins,col_pins** along with **row_num** and **col_num**	
Tell me one thing do you think it's possible to get the winner always. No, Sometimes conditions can come where all the parties get the same number of votes. So that time there should be conditions where we display **tie** or **no result.**	**ESR:** Varied!

```
Keypad keypad = Keypad( makeKeymap(keys), row_pins, col_pins, row_num, col_num);
```

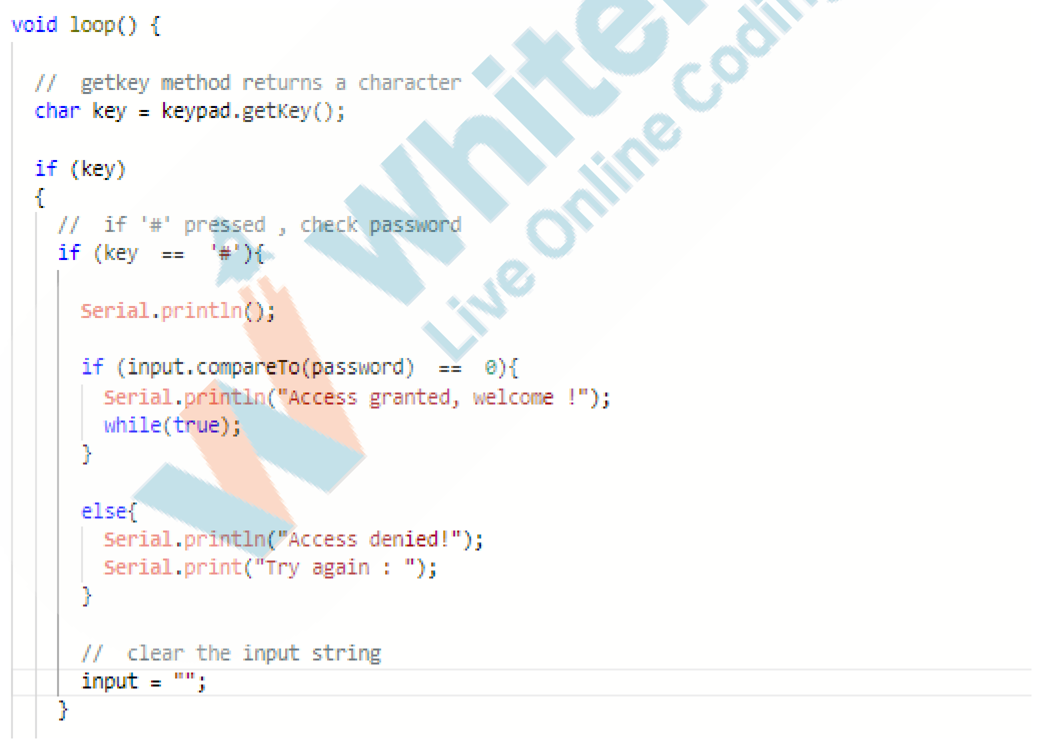
Initialize using **void setup()** function 3. **void setup()** is used to initialize 4. **Serial.begin() Serial.begin(9600)** is used for data exchange data speed. This tells the Arduino to get ready to exchange messages with the Serial Monitor at a data rate of 9600 bits per second.That's 9600 binary ones or zeros per second and is commonly called a **baud rate.** 5. **Serial.print** is used to print or display the user_input password on Serial Monitor	

```
void setup() {
  Serial.begin(115200);
  Serial.print("Enter password : ");
}
```

void loop() function is used to execute the main process. ● **Keypad.getKey()** returns the keycode of the	

pressed key, If the key is pressed return the keycode. The keycode is retrieved from the keymap array.

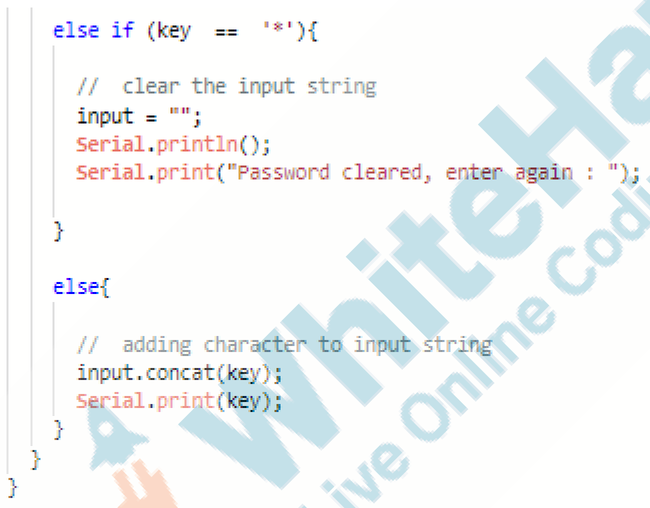
- **#** is used to check the password. After entering the six-digit password press the **#** key to check the correct password. compare method will check the user input with the store password.
- If it doesn't match with the password then it will display the Access Denied.
- if the input string matches with the password string print **Access granted, welcome!** else print **Access denied. Try Again!**



```
void loop() {

  //  getkey method returns a character
  char key = keypad.getKey();

  if (key)
  {
    //  if '#' pressed , check password
    if (key  ==  '#'){

      Serial.println();

      if (input.compareTo(password)  ==  0){
        Serial.println("Access granted, welcome !");
        while(true);
      }

      else{
        Serial.println("Access denied!");
        Serial.print("Try again : ");
      }

      //  clear the input string
      input = "";
    }
```

Now suppose the user entered the wrong password and the user wants to try a new password.

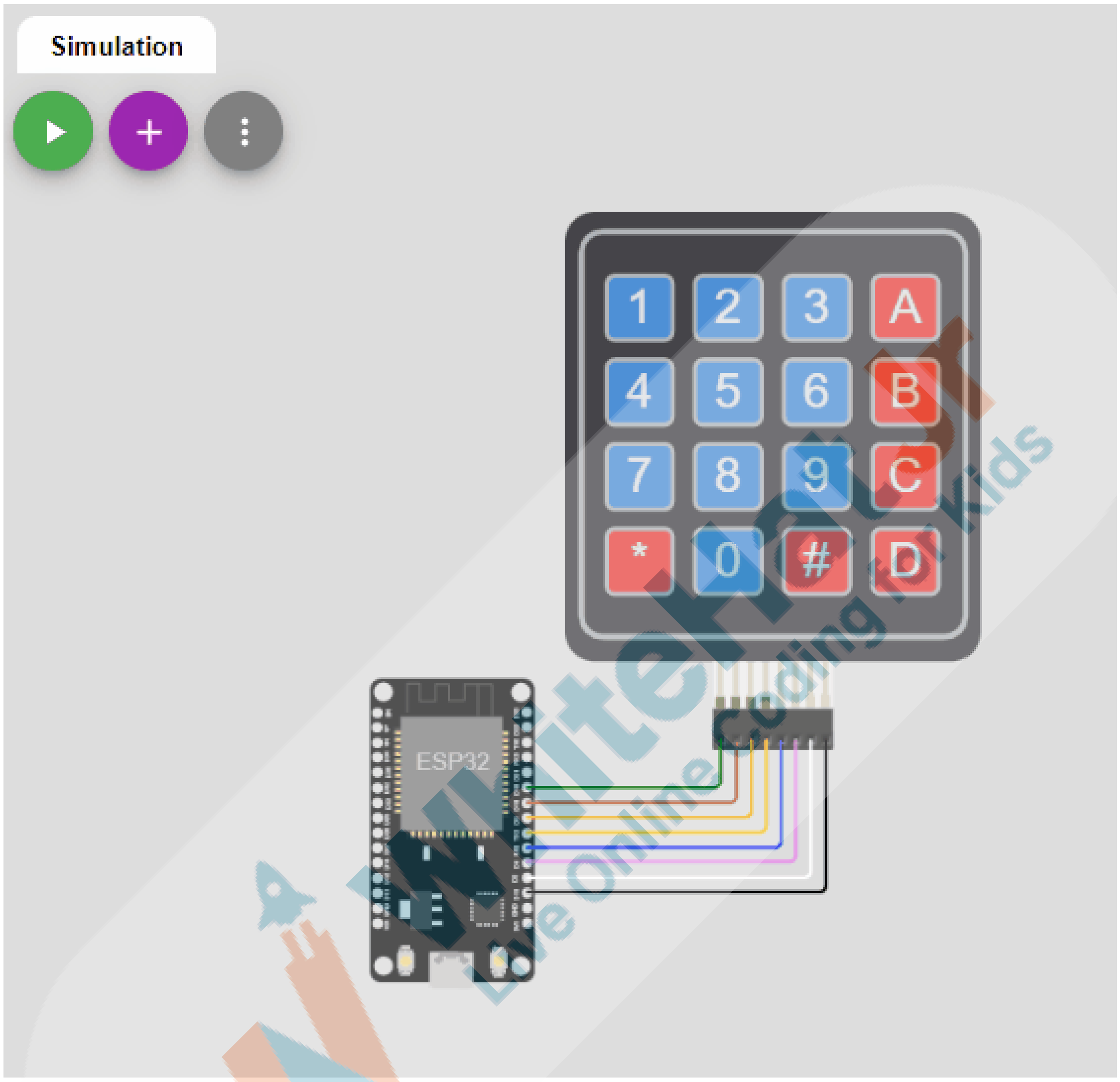
- To try password again press the * and write the password again.
- * will clear the **input** string and get ready to take another input from the user.
- **Serial.println()** is used to print the statement.After clearing it will display **(“ Password cleared, enter again”)**
- **concat** function will used to add the string.



```
        else if (key  ==  '*'){

          //  clear the input string
          input = "";
          Serial.println();
          Serial.print("Password cleared, enter again : ");

        }

        else{

          //  adding character to input string
          input.concat(key);
          Serial.print(key);
        }
      }
    }
```

Output:
Click on the Save button and then click on the simulation button

- Press the key and see the output on the Serial Monitor of the simulator.
- Just press the keys and you will get the output.



So, today we completed our security password application.	

Teacher Guides Student to Stop Screen Share

WRAP-UP SESSION - 05 mins

Activity details

Following are the WRAP-UP session deliverables:

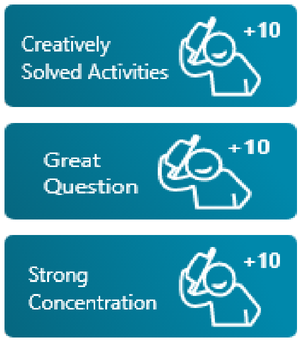
- Appreciate the student.
- Revise the current class activities.
- Discuss the quizzes.

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 about web servers	*Make sure you have given at least 2 hats-off during the class for:*  Creatively Solved Activities +10 Great Question +10 Strong Concentration +10

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	https://wokwi.com/
Teacher Activity 2	Keypad	https://docs.wokwi.com/parts/wokwi-membrane-keypad
Teacher Reference 1	Teacher Activity Reference Code	https://github.com/procodingclass/PRO-C259-Teacher-Activity
Teacher Reference 2	Reference Code	https://github.com/procodingclass/PRO-C259-Reference-Code
Teacher Reference 3	Project	https://s3-whjr-curriculum-uploads.whjr.online/7b6005b4-97fb-40db-9899-43cefe3b86a6.pdf

Teacher Reference 4	Project Solution	https://github.com/procodingclass/PRO-C259-Project-Solution.git
Teacher Reference 4	In-Class Quiz	https://s3-whjr-curriculum-uploads.whjr.online/62efb5ca-8066-4c8f-a7e8-180f94664d2a.pdf
Student Activity 1	Simulator	https://wokwi.com/
Student Activity 2	Keypad	https://docs.wokwi.com/parts/wokwi-membrane-keypad