SIT225 Data Capture Technologies

Pass Task: Hello Arduino!

Yasmin Pokia

S222245206

Task Objective

In this week, you have learned to write blink LED sketch for Arduino to periodically turn it ON/OFF and establish communication between Arduino and your computer through serial communication port. In this task, you are required to develop a communication protocol between Arduino and Python script running on your computer where Python script sends command to Arduino to blink LED a number of times and wait for Arduino to do the job and in response, Arduino sends a number back for which the Python script should wait that number of seconds before sending the next blink command and this process repeats.

Steps:

1. Python script runs and sends a random number through serial communication port where Arduino sketch is listening and receives the number. Python script should log the sending event in console with timestamp.

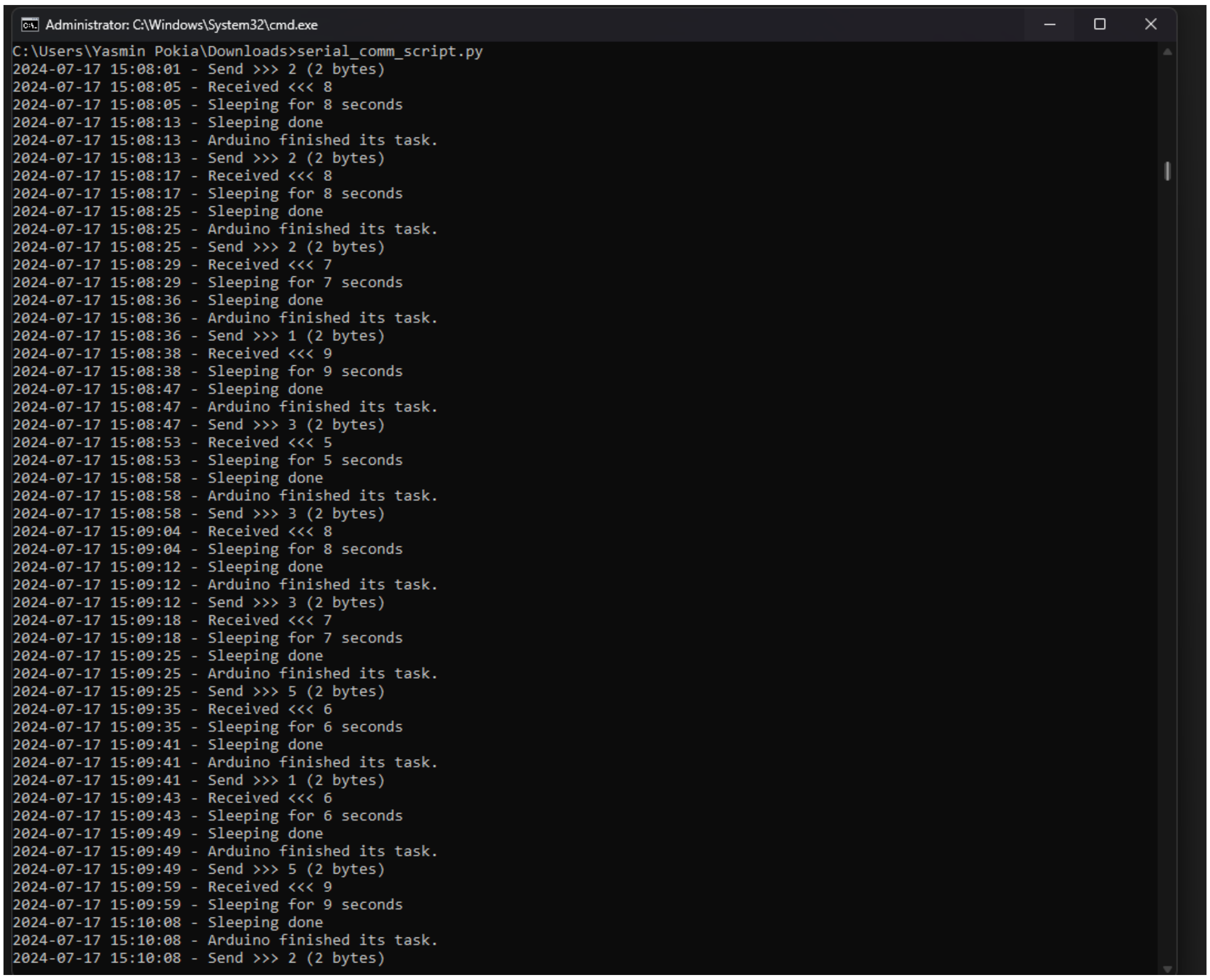
2. Arduino blinks the LED that number of times with 1 second interval.

3. Arduino writes to serial communication port a random number where the python script is listening and receives the number.

4. Python script receives the number, logs in console with timestamp and sleeps that number of seconds. Logs in console when sleeping is done.

5. Python script sends a number through the serial communication port same as step 1 and the process repeats forever.

Q2. Perform the task mentioned above in the Task Objective and keep running for few minutes. Capture the screen showing Python console output of communication log as evidence of interaction with Arduino sketch. Describe the log lines in the screenshot in a paragraph following the image.



```
C:\Users\Yasmin Pokia\Downloads>serial_comm_script.py
2024-07-17 15:08:01 - Send >>> 2 (2 bytes)
2024-07-17 15:08:05 - Received <<< 8
2024-07-17 15:08:05 - Sleeping for 8 seconds
2024-07-17 15:08:13 - Sleeping done
2024-07-17 15:08:13 - Arduino finished its task.
2024-07-17 15:08:13 - Send >>> 2 (2 bytes)
2024-07-17 15:08:17 - Received <<< 8
2024-07-17 15:08:17 - Sleeping for 8 seconds
2024-07-17 15:08:25 - Sleeping done
2024-07-17 15:08:25 - Arduino finished its task.
2024-07-17 15:08:25 - Send >>> 2 (2 bytes)
2024-07-17 15:08:29 - Received <<< 7
2024-07-17 15:08:29 - Sleeping for 7 seconds
2024-07-17 15:08:36 - Sleeping done
2024-07-17 15:08:36 - Arduino finished its task.
2024-07-17 15:08:36 - Send >>> 1 (2 bytes)
2024-07-17 15:08:38 - Received <<< 9
2024-07-17 15:08:38 - Sleeping for 9 seconds
2024-07-17 15:08:47 - Sleeping done
2024-07-17 15:08:47 - Arduino finished its task.
2024-07-17 15:08:47 - Send >>> 3 (2 bytes)
2024-07-17 15:08:53 - Received <<< 5
2024-07-17 15:08:53 - Sleeping for 5 seconds
2024-07-17 15:08:58 - Sleeping done
2024-07-17 15:08:58 - Arduino finished its task.
2024-07-17 15:08:58 - Send >>> 3 (2 bytes)
2024-07-17 15:09:04 - Received <<< 8
2024-07-17 15:09:04 - Sleeping for 8 seconds
2024-07-17 15:09:12 - Sleeping done
2024-07-17 15:09:12 - Arduino finished its task.
2024-07-17 15:09:12 - Send >>> 3 (2 bytes)
2024-07-17 15:09:18 - Received <<< 7
2024-07-17 15:09:18 - Sleeping for 7 seconds
2024-07-17 15:09:25 - Sleeping done
2024-07-17 15:09:25 - Arduino finished its task.
2024-07-17 15:09:25 - Send >>> 5 (2 bytes)
2024-07-17 15:09:35 - Received <<< 6
2024-07-17 15:09:35 - Sleeping for 6 seconds
2024-07-17 15:09:41 - Sleeping done
2024-07-17 15:09:41 - Arduino finished its task.
2024-07-17 15:09:41 - Send >>> 1 (2 bytes)
2024-07-17 15:09:43 - Received <<< 6
2024-07-17 15:09:43 - Sleeping for 6 seconds
2024-07-17 15:09:49 - Sleeping done
2024-07-17 15:09:49 - Arduino finished its task.
2024-07-17 15:09:49 - Send >>> 5 (2 bytes)
2024-07-17 15:09:59 - Received <<< 9
2024-07-17 15:09:59 - Sleeping for 9 seconds
2024-07-17 15:10:08 - Sleeping done
2024-07-17 15:10:08 - Arduino finished its task.
2024-07-17 15:10:08 - Send >>> 2 (2 bytes)
```

The communication between the Arduino and the Python script is recorded in the log. Every log entry begins with the action completed and is followed by a timestamp. The Arduino receives a random integer (Send >>> x (2 bytes)) from the Python script, where x is the provided integer. After processing this integer, the Arduino waits x seconds, blinks the LED x times, and then returns a random integer (Received <\\ y), where y is the integer that the Python script received. After logging the receiving integer, the Python script pauses for x seconds before iterating through the loop once more. The log entry indicating that "Arduino finished its task." verifies that the Arduino has done flashing and is ready for the Python script to submit the next number. This constant communication keeps the Python script and the Arduino in sync, proving that the given task was successfully completed.

Q3. Paste Python and Arduino sketch and explain program steps. Your explanation should match the description you prepared for Q2.

Python Program Explanation:

```python
import serial
import random
import time
from datetime import datetime

# Set baud rate, same speed as set in your Arduino sketch.
```

```python
baud_rate = 9600

# Set serial port as suits your operating system
s = serial.Serial('COM5', baud_rate, timeout=5)

def log(message):
    print(f"{datetime.now().strftime('%Y-%m-%d %H:%M:%S')} - {message}")

while True:  # Infinite loop, keep running
    # Generate a random integer between 1 and 5.
    data_send = random.randint(1, 5)

    # Write the generated integer to the serial port, encode it as UTF-8.
    d = s.write(bytes(str(data_send) + '\n', 'utf-8'))
    log(f"Send >>> {data_send} ({d} bytes)")

    # Read from the serial port until "DONE" is received.
    while True:
        d = s.readline().decode("utf-8").strip()
        if d == "DONE":
            log("Arduino finished its task.")
            break
        if d.isdigit():
            wait_time = int(d)
            log(f"Received <<< {wait_time}")

            # Sleep for the received amount of seconds.
            log(f"Sleeping for {wait_time} seconds")
            time.sleep(wait_time)
            log("Sleeping done")
```

1. Serial Initialisation and Logging: A serial connection (s) to COM5 are initialised by the Python script at a baud rate of 9600. It specifies the log function, which prints timestamps along with messages.
2. Main Loop Execution: To maintain constant communication with the Arduino, the script enters an endless loop (while True):

- Create Random number: It creates a data_send random number, ranging from 1 to 5.
- Data Send to Arduino: This action is logged and a random integer is sent to the Arduino via the serial port.
- Waiting for Arduino Response: It goes into another loop to read data from the serial port until it gets the "DONE" signal from the Arduino, which means that the task has been completed. When a number is input, the Arduino understands it as the amount of time (in seconds) that it will blink its LED and go to sleep.
- Handle Arduino Response: The script logs a numeric value as "Received" and then goes to sleep for the specified number of seconds after getting it.

3. Logging: The script records all of the actions that are done during the process, such as transmitting data, getting acknowledgements, and sleeping.

Arduino Sketch Explanation:

```
int ledPin = 13; // Set the LED pin (e.g., built-in LED on pin 13)
int x;

void setup() {
  Serial.begin(9600);  // Set baud rate
  pinMode(ledPin, OUTPUT); // Set the LED pin as output
}

void loop() {
  while (!Serial.available()) {} // Wait for data to arrive

  // Read string data from Serial and ignore newline/carriage return
characters
  String input = Serial.readStringUntil('\n');
  x = input.toInt();

  // Debugging print to ensure the correct value is received
  Serial.print("Received: ");
  Serial.println(x);

  // Wait for x seconds before blinking the LED
  for (int i = 0; i < x; i++) {
    delay(1000);  // Wait for 1 second each loop iteration
  }

  // Blink the LED 'x' times
  for (int i = 0; i < x; i++) {
    digitalWrite(ledPin, HIGH);   // Turn the LED on
    delay(500);                  // Wait for half a second
    digitalWrite(ledPin, LOW);    // Turn the LED off
    delay(500);                  // Wait for half a second
  }

  // Send a random number back to Python
  int response = random(5, 10);
  Serial.println(response);

  // Indicate that the Arduino is ready for the next command
  Serial.println("DONE");
  Serial.flush();
}
```

1. Initialisation: To initialise, the Arduino configures pin 13 as an LED output and establishes serial communication at a rate of 9600 baud.
2. Main Loop Execution: The loop() method on the Arduino runs continuously:

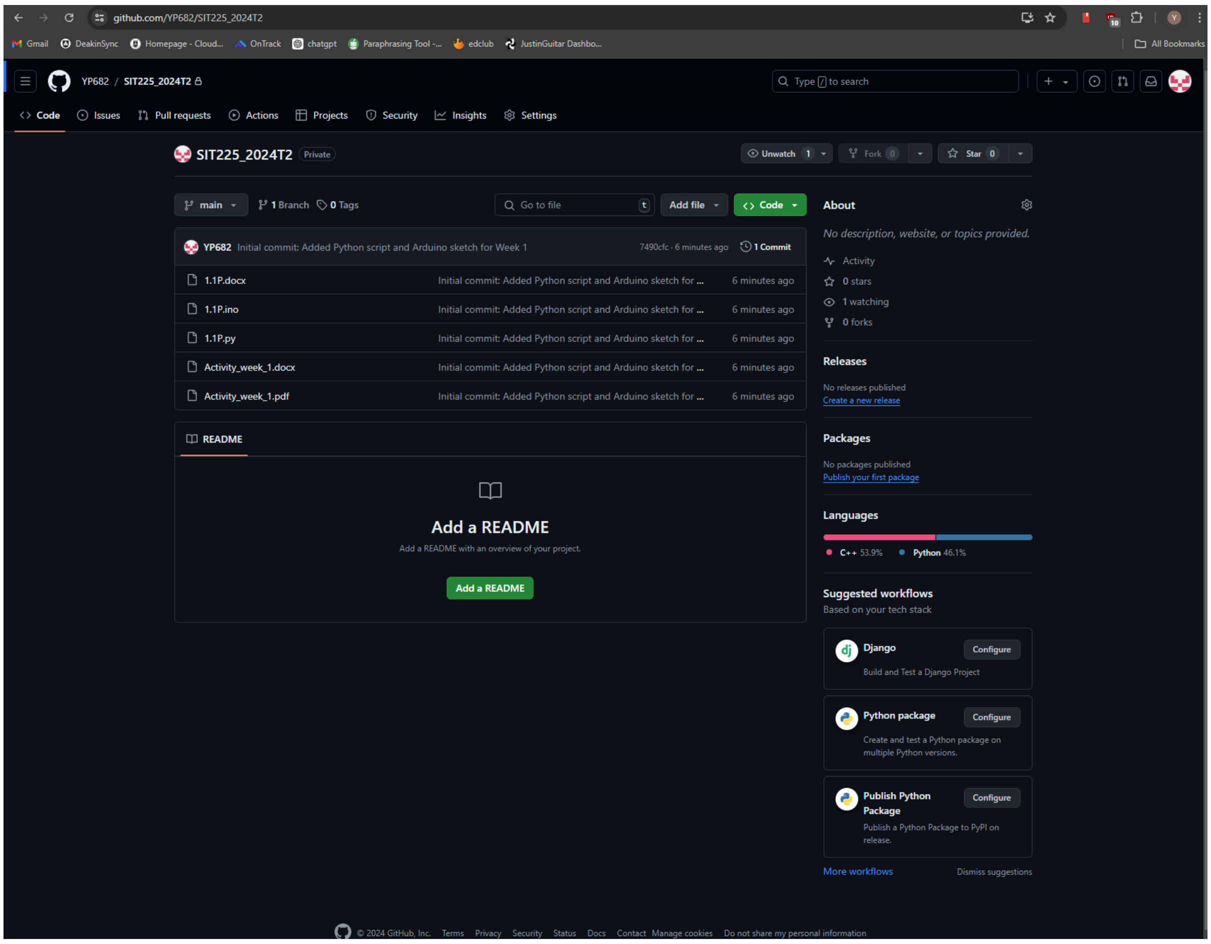
- Waiting for Serial Data: This process waits for the serial port to receive data (Serial.available()).
- Read and Parse Data: Until it comes across a newline (\n), it reads the incoming string from the serial port. It uses toInt() to convert this string to an integer (x).
- Debugging Print: To aid with debugging, it prints the integer that was received.
- Delay Before Blinking: The LED doesn't start to blink for x seconds.
- Blink the LED: It turns the LED on and off with half-second delays (delay(500)) by blinking the LED x times.
- Send Response to Python: Following its blinking, it creates a random integer (response) between 5 and 10 and serially communicates it back to the Python script.
- Signal Completion: It sends "DONE" to the Python script to let it know that the blinking is complete and that it is prepared to receive another command.
- Make sure all data is transmitted out of the serial buffer by using the flush serial buffer function (Serial.flush()).

Q4. Create a video in Panopto/Cloud Deakin showing your program execution and any instruction to be followed in order to run your code, share the link in your report.

https://deakin.au.panopto.com/Panopto/Pages/Viewer.aspx?id=a2b88143-e96c-4e33-a7fd-b1b00080a122

Q5. Create a directory 'SIT225_2024T2' in your drive and a subdirectory 'week-1' where you copy the Python script file and Arduino sketch file. Create a new private repository on GitHub sharing 'SIT225_2024T2' folder. You will be creating new subfolders every week. Include the link to your repository in your report with a GitHub page screenshot of weekly folder content. A tutor may try to access your GitHub link, if necessary. Give access to your tutor by adding tutor's email address as a collaborator of your private repository.

https://github.com/YP682/SIT225_2024T2



SIT225_2024T2 Private

1 Branch 0 Tags

YP682 Initial commit: Added Python script and Arduino sketch for Week 1 — 7490cfc · 6 minutes ago — 1 Commit

1.1P.docx	Initial commit: Added Python script and Arduino sketch for ...	6 minutes ago
1.1P.ino	Initial commit: Added Python script and Arduino sketch for ...	6 minutes ago
1.1P.py	Initial commit: Added Python script and Arduino sketch for ...	6 minutes ago
Activity_week_1.docx	Initial commit: Added Python script and Arduino sketch for ...	6 minutes ago
Activity_week_1.pdf	Initial commit: Added Python script and Arduino sketch for ...	6 minutes ago

Add a README

Add a README with an overview of your project.

About

No description, website, or topics provided.

Activity
0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

C++ 53.9% Python 46.1%

Suggested workflows

Based on your tech stack

Django — Build and Test a Django Project

Python package — Create and test a Python package on multiple Python versions.

Publish Python Package — Publish a Python Package to PyPI on release.

© 2024 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my personal information