Schedule

Pre-course form to understand the coding proficiency and ease with hardware of the students participating, accordingly split them into groups.

WEEK 1

1. breaker (fun) + What’s code, why do we need it. Teach basics of data structures- strings, arrays, including ascii code+ accessing elements/characters.
2. Doing array, string manipulation exercises + introducing encryption historically
3. Combining encryption and data structures

Exercises, videos- homework : remain busy

WEEK 2

1. Setting up- live demo of how to download, explain what esp32s are
2. Mesh networks day 1- Basic activities on the mesh.
3. Mesh networks day 2- Using mesh networks for encrypted communication, how does this relate IRL
4. Tangible Output + presentations

Till 17th- complete lecture work (ppt, visuals, etc.) + script for downloads   
Roshni - Lec. 1-4  
Tushar- Lec. 5-7 and challenge cards

Lecture Plans- G1 (9th-10th)

Allow decision points - flexible.   
Condense 2nd week setting up parts into 1st week if possible.  
Try to do more low-effort, high-impact activities.  
  
Use the weekends to send out emails on interesting auxiliary things that people can look into.  
1 hr OH- where people come with doubts on a document, come a little prepared.   
  
2nd edition: 9th, 9th and 10th, 11th and 12th.  
Focus on recall moments - think of the flow - explain why you are doing something, make it clear what the motivation is, set context in the beginning.

Content similar, structure it differently- needs to be more engaging not module by module. Go through the presentation topics.   
  
Tangible outputs- something they can take home with them or they can keep   
  
What are we adding: concepts will sink in more and students will be able to recall because of how we are integrating it with other.  
  
  
**Use VScode asap - allow time for personalization   
GitBash - download it using the script**

**Jupyter notebook inside VScode in place of colab  
Package managers + jupyter notebook - update script, ensure consistency**

**CLASS 1**

**CTRL+YOU: Taking Back Power in the Digital World**

**(10 mins to settle)**

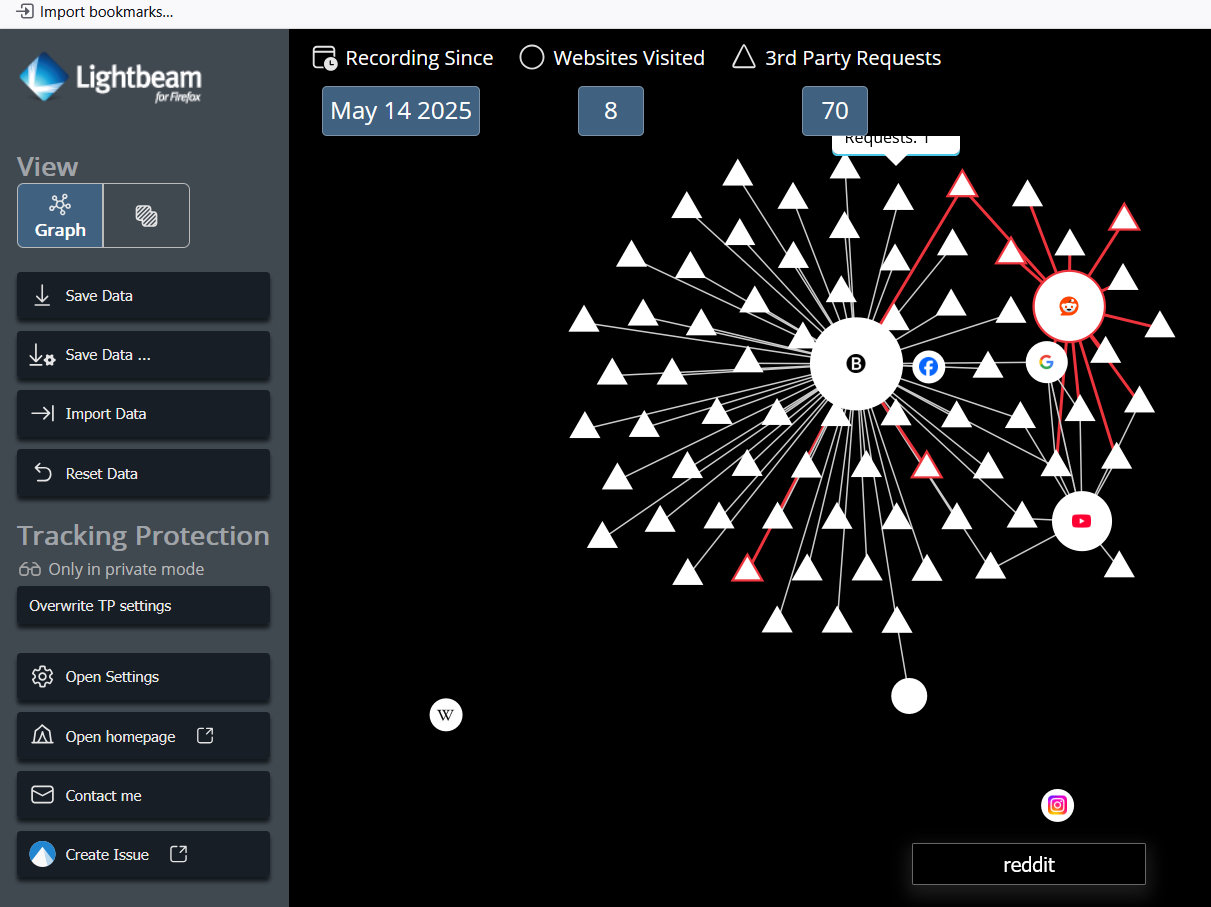
**Ice breaker: Tissue paper introduction / some other introduction 20 mins (up-to 3)**

**Digital Awakening - 25 mins**

*"How the Internet Sees You"*Kick off with a fun, slightly shocking look at data trails, cookies, targeted ads, etc.

When you visit a webpage, what does “accept all cookies mean”?

**Activity: Using privacy badger for students and lightbeam for us (show the third-party mappings)**

****

**What is lightbeam -** [**https://www.youtube.com/watch?v=PvqGy9wz\_wA**](https://www.youtube.com/watch?v=PvqGy9wz_wA)

**How do you use lightbeam -** [**https://www.youtube.com/watch?v=4A7e6bOBkQM**](https://www.youtube.com/watch?v=4A7e6bOBkQM)

**Explain what is happening-> Third party trackers, cookies, what information are they trying to access?**

**What are some peculiar things that you notice? - reason what you see   
Instagram, Facebook unexpectedly have low no. of third party trackers   
First-party interconnections**

**Trailing question: Is it possible to use the internet without being tracked?**

This is just the beginning… Over the course of the next six sessions we will think more deeply through certain guiding questions about how we interact with the internet.

**Explain logistics: 15 mins**

1. Get a sense of the level of familiarity with coding- accordingly pace the first class   
   (coding basics heavy OR discussion heavy)
2. Schedule for the next few sessions + why we will be doing every step
3. Challenge cards - activities for those who are running ahead (there might be rewards for some of these)
4. Attendance
5. Installations and using Jupyter
6. Deliverables - reflections
7. Introduce as Splitting into groups (9th-10th and 11th-12th)

**Appropriate Bridge**

"Pretty wild, right? You just saw how many third-party sites are tracking your activity — and most of us had *no idea* that was happening.

But here’s the thing — that tracking doesn’t just *magically* happen. It’s all made possible through **code**.

The websites you visit, the trackers that follow you, the pop-ups that ask for cookies — all of that runs because someone wrote code to make it work.

So if you've ever wondered *how apps know what you like, or how websites ‘remember’ you* — now’s the time to peek under the hood.

Let’s first break down a few coding basics so that we are all on the same page.

| **GROUP 1** | **GROUP 2** |
| --- | --- |
| **Coding explainers (~40 mins)** | **Coding explainers (~20 mins)** |
|  | **Revision (~20 mins)** |
| **Installations (~20 mins)** | **Installations (~20 mins)** |

Coding basics: (~ 40 mins) - do this live on google colab and follow alongside me   
What is code  
Program structure   
What is a function, variables, comments

**CLASS 2**

**Logistics: Projector, ~ 9 boxes, with number passcodes, printouts of the instructions.**

**HIGH (20 mins)**  -

5 mins to explain, 15 mins to solve the puzzle

game where the 20/25 students are split into groups. Each of them are handed a puzzle box with a passcode lock. They need to solve multiple rounds of puzzles with basic forms of encryption to get the final passcode and open the box. We can keep this timed as well- say 20 minutes. (45/9)= 5 people per group, 9 groups total and 4 groups per GROUP.

Remember that everything I will be teaching you now will be used to solve a really cool challenge at the end of the class…

Speed up or down on the basis of how much people have coded before

**VISUAL EXPLAINER + DISCUSSION (15 mins)** - Use the activity to explain strings, accessing characters. We can walk through (using visual explainers) /have a discussion on how students went about trying to crack the code so that they have a mental sense of what goes into thinking about manipulating characters. - 20 minutes. Unicode for characters.

**ARRAY EXPLAINER (20 mins)** - Use a visual explainer of boxes/doors that are indexed with an object behind them. Accessing these objects (apples, cars, cats etc.). Show how we would swap them, changing objects behind a door, etc.

**Activity to prevent distraction - keep it short also nudge students to use challenge cards if they are already familiar with what is being taught/done with their exercises (5 mins)   
  
JUPYTER NOTEBOOK CODING WITH AN INTERESTING THEME (45 mins)** - Extract characters, swap positions of elements, get a substring, introduce some functions. Make the colab slightly repetitive so that they get the hang of what certain functions do.  **THEME:   
  
Link back to previous day’s activity: should companies be allowed to use your data to improve services, if they don’t sell it? (15 mins)**

(i) Can "improving your experience" also mean subtly nudging your behavior to serve the company’s goals (e.g., keep you scrolling, make you buy more)? What other concealed motives may a company use this data for?

How would you draw the line between helpful customization and manipulation?

(ii) Can true consent exist in the current digital world, or are we just accepting terms we don’t understand?

If you can’t access a service without agreeing to data collection, is that coercion or fair exchange?

What would meaningful, informed consent look like for a teenager using a new app?

**Exercise Questions gist:** The business of you: who gets to profit from your data? **Mimic a malicious actor trying to access people’s data…**

Access from arrays (names, private conversations, their purchase history, their current interests) - array accessing   
  
Substrings  
  
Accessing characters

**Trailing Q: As a crafty, self-serving capitalist, what might you do with this data that you have? What kind of ads might you target?**

**Readings:**

**Cambridge Analytica scandal- politics  
Manipulation without actual consent - ethic**

**CLASS 3**

**Logistics: Projector  
  
Last session we discussed the ethics of companies using our data   
  
THEME: (15-20 mins)**

**Do you think media online should be policed/controlled? If so, by who- governments, developers, or users?**

(i) What would be the implications of there being no regulation on online content vs. it being heavily regulated by a particular party? - unhindered freedom of speech at the cost of being insensitive. Hindering freedom of speech at the cost of selective voices being heard.   
  
(ii) How can we try to strike a balance in terms of authority? Who are all the stakeholders involved - i.e. who all will be affected by such a decision?

**HIGH (10 mins) -** Activity that highlights the importance of writing short code for something that you want to do over and over.   
  
Call on students to perform a funny set of actions multiple times-   
e.g. if your name starts with A jump five times else shake hands 4 times AND wave 5 times, etc.   
  
Ask them how we could reduce these sets of statements.   
  
EXPLAINER(20 mins)- Loops

**LOOPS AND IF ELSE (1 hr)-** COLAB, with storyline.   
  
Solidify concepts of loops, if-else, arrays, and strings and their manipulations.

### **Coding Exercises:** Silenced by code: Who gets to speak?

1. **Moderation Simulator:** Write a function to flag “bad” content using a banned words list.  
    *[List of strings + string containment]*
2. **Voting on Content:** Users vote “keep” or “remove” on a post — return result based on majority - striking balance in terms of authority   
    *[Arrays + counting votes]*
3. **Custom Thresholds:** Allow users to define how many flags it takes to hide content.  
    *[If statements with user input]*

**Supporting readings:**   
Rohingya- poor content moderation effects   
#MeToo movements being shut down in China - censorship effects

(30 mins free here)

**CLASS 4**

**Reflection point:**

**WHAT IS ENCRYPTION? (20 mins) -** Introduce simple ciphers: Caesar, Atbash, ROT13 - class tries out examples in between   
  
How exactly is encryption different from encoding?   
  
**HIGH (20mins)-** Show how shifting characters work. Draw real-life parallels (WWII, secret messages, WhatsApp encryption, SIM card (CIA story)).

**DESIGNING YOUR OWN ENCODING (1 hr 10 mins):** Part of final day presentations and can be worked on over the weekend.  **Describe your algorithm, what is the element of secrecy?  
  
Examples of encoding/encrypt**

**Reverse strings   
Emoji-to-letter mapping  
  
  
Reading:**

[**Asymmetric Key Cryptography | RSA Encryption Algorithm | Asymmetric Encryption | Simplilearn**](https://www.youtube.com/watch?v=sH74nrEefyI)

**Symmetric encryption:**

[**Lesson 14: The Vigènere Cipher**](https://www.youtube.com/watch?v=e9bXWM7VATc)

**Asymmetric encryption:**

[**Public Key Cryptography: RSA Encryption**](https://www.youtube.com/watch?v=wXB-V_Keiu8)

[**Diffie-Hellman Key Exchange**](https://www.youtube.com/watch?v=YEBfamv-_do) **- Colour mixing can be used to show how**

[**11 Cryptographic Methods That Marked History: From the Caesar Cipher to Enigma Code and Beyond**](https://interestingengineering.com/innovation/11-cryptographic-methods-that-marked-history-from-the-caesar-cipher-to-enigma-code-and-beyond)

**Class 5 (Installation day) - Possibly do this as a combined class with + 1 volunteer needed   
  
Disclaimer : Let them know that they are installing**

**Logistics:** Projector, Type C to C cables or Mac multiport adapter

**ESPs and Microcontrollers (10 mins):** Introduce ESP32s and microcontrollers in general - what they are, why, when and how to use them.

Documentations send over

**Discussion: (~ 30 mins)**The internet may appear invisible but what actual infrastructure goes into making it work? + how do we send messages across the internet - link to next day’s class.

How do we communicate with such infrastructure + how do we communicate with our laptops   
  
(decentralization? - blockchain possibly?)

**Terminal and the GitHub Repo (10 mins):** Introduce the Github Repo and the terminal. Quick basics of what they are and why we are using them.

**Setting up the ESPs and the terminal (1 hr)**: live demo of the setting up so people can follow along and two volunteers helping students.

Guiding them through the repo and having them setup all the necessary software, then getting them to connect the ESPs and testing the setup.

**Optional activity if everyone done early**

**10 mins buffer / Questions / Open Stage**

**Class 6**

**Logistics:** Projector, LEDs

**Introduction to Networks (30 mins):** What is a network? What is a mesh network? How are we using the ESPs to create the network and communicate through it?

**Activities on the Mesh (1 hrs):** Follow the GitHub repo and have them perform the activities: sending messages across the network to each other.

1. Set up the ESPs.
2. Map out the mesh network and figure out what all devices are connected immediately to theirs.
3. Light up their own LEDs first. **The firmware has to be edited for this. #todo**
4. Send a message to a specific node and figure out whose all LEDs will light up in between.   
     
   E.g. A, B, C, D, and E are seated randomly around the room. We ask everyone to send a message to C and figure out whose all LEDs will light up in the process. C could try figuring out if A were to send them a message, what all LEDs will light up.  
     
   **#Challenge Card**: what could they do to avoid sending a message to someone in between. Say, A sending a message to C means B lighting up first. So, what can A do to prevent B from lighting up and directly sending a message to C.   
     
   Deepraj had suggested we put down some logic, sort of flags, wherein when the signal reaches a node, the logic verifies if the flag = true, if so light up, else ignore and move on. **#todo**

**Fish and net activity to get attention (~5 mins)**

**Ideation point-** Now that you’ve discovered the features of the mesh network - (decentralized communication, no need of internet connection, passing a message but everyone in between can read it, etc.)  
`  
What circumstances might you think a mesh network is useful, what are aspects that can be improved on and to what end?

**Class 7**

**Logistics:**

**Primary Mesh Network Activity (~1 hr):**

**End-to-end encryption**

**Short audio - message sent!**  
  
**Morse code based activity-** Light up LEDs connected to ESPs (if possible) to create morse code.   
  
**Network tag-** One ESP is "It" and sends a tag message to a random node. That node becomes "It" and continues the chain. LED flashes when you're "It" or when you get tagged. Add a timer: Who's "It" at the end?

Lecture Plans- G2 (11th-12th)

More challenging challenge cards  
  
**Class 1**Coding basics: (~ 40 mins)   
What is code  
Program structure   
What is a function, variables, comments

(~ 15 mins quick explainers) + (~25 mins to solve exercises)   
  
**Class 2**

Time for array, string explainers shorter Jupyter notebooks make little more advanced (replace a few questions with harder ones**)   
  
  
Class 3   
  
HIGH (10 mins) -** Activity that highlights the importance of writing short code for something that you want to do over and over. Similarly, something to show importance of if else statements  
  
Call on students to perform a funny set of actions multiple times-   
e.g. if your name starts with A jump five times else shake hands 4 times AND wave 5 times, etc.   
  
Ask them how we could reduce these sets of statements.

(adapt to something not bookish but assumes some knowledge)   
  
Coding exercises more advanced   
  
**Class 4**

**WHAT IS ENCRYPTION? (20 mins) -   
  
Introduce caesar, ROT13, Atbash - Why is it a laughable form of encryption - how would you suggest improving it? (15 mins)**How exactly is encryption different from encoding? (5 mins)   
  
**HIGH (20mins)-** Show how shifting characters work. Draw real-life parallels (WWII, secret messages, WhatsApp encryption, SIM card (CIA story)).  
  
Play with an enigma machine online - what do you observe, explain it better. What are still some pitfalls?  
  
Modern day encryption

Class 2- Puzzle Box

## **The Cipher Box Challenge**

**Participants**: 9 groups of 4–5 students  
 **Duration**: 20 minutes  
 **Objective**: Open your team’s box by solving 4 logic-based puzzles. Each correct answer gives you 1 digit of the final 4-digit code.

### **Setup**

Each group receives:

* A sealed puzzle box or envelope with a 4-digit lock (real or symbolic)
* 4 envelopes or sheets labeled Round 1 to Round 4 containing the puzzles
* Optionally: a leaderboard or visible countdown clock

## **Puzzle Rounds (No Prior Coding Required)**

Each round is designed to simulate the mental logic used in string and array manipulation without using programming terminology.

### **Round 1: Letter Trail**

**Concept**: Accessing a character by position, alphabet mapping

**Puzzle Instructions**:  
 You are given the word: **"EXPLORATION"**

1. Find the 5th letter in the word.
2. What letter comes directly after it in the alphabet?
3. What is the alphabetical position of that new letter? (A=1, B=2, ..., Z=26)
4. If the number is greater than 9, reduce it to a single digit by adding the digits together.
5. This is your first digit.

### **Round 2: Slice of Meaning**

**Concept**: Breaking a sentence into components, accessing by position

**Puzzle Instructions**:  
 You are given the sentence:  
 **"Digital footprints never disappear."**

1. Count how many words are in the sentence.
2. Use that number to select the corresponding word (e.g. if there are 5 words, use the 5th word).
3. Count how many letters are in that word.
4. If the number is greater than 9, reduce it to a single digit.
5. This is your second digit.

### **Round 3: The Grid Map**

**Concept**: Understanding coordinates in a 2D structure

**Puzzle Instructions**:

You are given a 4×4 letter grid:

A R T S

B I N G

D A T A

C O D E

You are also given these coordinates:  
 (0,1), (1,2), (2,3), (3,0)

1. Use the coordinates to extract letters from the grid. Each pair represents (row, column).
2. Combine the letters into a sequence.
3. Count how many vowels appear in the sequence.
4. This count is your third digit.

### **Round 4: The Pattern Key**

**Concept**: Recognizing and extending patterns, alphabetical mapping

**Puzzle Instructions**:

You are given the letter sequence:  
 **"F H J L N"**

1. Identify the pattern in the sequence. What’s happening between the letters?
2. Determine the next letter that would follow in the sequence.
3. Find the position of that letter in the alphabet. (A=1, ..., Z=26)
4. If the number is more than 9, reduce it to a single digit by adding the digits together.
5. This is your fourth digit.

### **Final Step**

Each group combines their four digits to form a 4-digit code.  
 Once they have the code, they attempt to unlock their box. The first team to open their box wins.

Inside the box, you can include:

* A clue or artifact for the next part of the workshop
* A symbolic “data key” or printed reward
* Something that connects back to your digital identity theme

**PRINTABLE VERSION**

**The Cipher Box Challenge: Puzzle Cards**

**Round 1: Letter Trail**

You are given the word: EXPLORATION

1. Find the 5th letter in the word.
2. What letter comes directly after it in the alphabet?
3. What is the alphabetical position of that new letter? (A=1, B=2, ..., Z=26)
4. If the number is greater than 9, reduce it to a single digit by adding the digits together.
5. This is your first digit.

**Round 2: Slice of Meaning**

You are given the sentence:  
 "Digital footprints never disappear."

1. Count how many words are in the sentence.
2. Use that number to select the corresponding word (e.g. if there are 5 words, use the 5th word).
3. Count how many letters are in that word.
4. If the number is greater than 9, reduce it to a single digit.
5. This is your second digit.

**Round 3: The Grid Map**

You are given a 4x4 letter grid:

A R T S  
 B I N G  
 D A T A  
 C O D E

You are also given these coordinates:  
 (0,1), (1,2), (2,3), (3,0)

1. Use the coordinates to extract letters from the grid. Each pair represents (row, column).
2. Combine the letters into a sequence.
3. Count how many vowels appear in the sequence.
4. This count is your third digit.

**Round 4: The Pattern Key**

You are given the letter sequence:  
 "F H J L N"

1. Identify the pattern in the sequence. What’s happening between the letters?
2. Determine the next letter that would follow in the sequence.
3. Find the position of that letter in the alphabet. (A=1, ..., Z=26)
4. If the number is more than 9, reduce it to a single digit by adding the digits together.
5. This is your fourth digit.

**Final Step: Unlock the Box**

Combine the four digits you discovered in each round.  
 This 4-digit code will open your team’s puzzle box.  
 Good luck!

Complicated Version

**Class 2 - Puzzle Box**

**The Cipher Box Challenge: Puzzle Cards**

**Round 1: Shifting an Alphabet**

You are given the word: mAkErSpAcE

1. Find the 7th letter in the word.
2. Shift alphabetically forward that 7th letter by the number of unique vowels in the entire word.
3. Convert the letter to ASCII code. *(Look up what they are.)*
4. Count the number of unique consonants in the word and add it to the ASCII code.
5. Add the digits of the final number until you get a single digit number.
6. That’s your first digit!

**Round 2: Slice of Meaning**

You are given the sentence:  
 "Cyber security is a shared responsibility, not just an IT concern."

1. m = (count the number of words with less than five letters)  
   n = (count the number of words with more than five letters) + 1
2. Reverse the **n**th word in the sentence.
3. Remove all the vowels from the **m**th word.
4. Add the new **n**th word and the new **m**th word. *(Can strings be added? Reason!)*
5. Count the number of letters.
6. Add the digits of the final number until you get a single digit number.
7. That’s your second digit!

**Round 3: The Grid Map**

You are given a 6x4 letter grid:

T R A C E   
H A C K S   
O N L I N   
E D A T A   
P A S S W   
O R D S !  
You are also given these coordinates:  
(0,0), (1,2), (2,3), (3,1), (4,4), (5,0), (2,0), (3,3), (5,2)

1. Use the coordinates to extract letters from the grid. Each pair represents (row, column).
2. Combine the letters into a sequence.
3. Reverse the string.
4. Count the number of unique consonants in the reversed string.
5. Add the digits of the final number until you get a single digit number.
6. That’s your third digit.

Were any of the steps here unnecessary?

**Round 4: The Pattern Key**

You are given the letter sequence:  
 "B E H K N"

1. Identify the pattern in the sequence.
2. Determine the next letter that would follow in the sequence.
3. x = the position of that letter in the alphabet. (A=1, ..., Z=26)
4. y = number of letters in the given sequence  
   z = y mod x
5. And, that’s your fourth digit!

**Final Step: Unlock the Box**

Combine the four digits you discovered in each round.  
This 4-digit code will open your team’s puzzle box.  
Good luck! :)

Details of YSP

1. 45 students in total in the first edition (2 weeks: from the 19th to 30th May).
2. 64 students in total in the second edition (2 weeks: from the 2nd to 13th June).
3. Schedule: <https://drive.google.com/file/d/1A315FI3pCy3rVd-KPY_YevkQRpjT8kMe/view>

To work on

FOR CLASS 1

Fit in time for installations

1. Supporting readings based on the discussions
2. Presentation (opening)
3. Basic coding explainers + gamified practice (G1 and G2)

FOR CLASS 2

1. Puzzles to open the box
2. Visual explainers for strings, arrays.
3. Theme based presentation- questions for discussion
4. Jupyter notebook encompassing string and array manipulation - with theme + more advanced ver. For G2
5. Readings

FOR CLASS 3

1. Presentation - discussion
2. Decide activities to show importance of loops, if-else
3. Visual explainers for loops, if-else
4. Jupyter notebook- loops, if-else, strings, arrays
5. Readings
6. Snapshot of mesh network - Deepraj
7. Day 1 presentation

Other low-effort things

1. Ice breaker - tissue roll
2. Set up Acadly
3. Set up firefox lightbeam for ourselves

* Flexibility
* Low-effort, high-impact
* Guiding question for day 1
* Non-linear

Tasks for us

1. we need to add another command in the command\_interface file: where they light up their own LEDs through the hex codes, i.e. they send a message to themselves.
2. the device\_list.py needs to be updated (allowed the ESPs)
3. talking about the computer - box: input (keyboards), output; bring the box diagram back when introducing the microcontroller. the inputs and outputs as the pins. So, catchphrases and analogies.
4. Think of the background work that will go into each element. Reduce that effort.
5. HIGH point: humor?
6. make sure they dont get lost as to why they are doing something – motivation has to be there; set context first and then proceed to the activities.
7. section by section models dont really work. so use these sections as elements to pick and move around as the workshop flows.
8. critically thinking about the ideas they are taught in the class – instead of having the sessions be tutorials, but critical engagement – how those affect their daily life

day 1: contextualize

after day 7: leave them with an idea to keep thinking about; engaging them with some questions/ideas

1. visual cryptography as an activity (challenge cards (for those who are running ahead)): figure out what software we used to make this and come with a new print for us to see

1: context

2,3: introduction

3,4,5: writing the code, colab

4,5: github repo download

day 3 – maximum engagement with coding and stuff

4 and 5 – work deescalate in terms of the effort they put – 5 the full network should be up. 4 and 5 they do that.

eeeeeeeeeeeeee

2 hrs have to go for the github repo and the terminal setup

—

1. **Create the bash script for both ubuntu and macos**

—

1. ask deepraj how the kids behaved last year – whether they were getting distracted or bored despite the challenge cards or class activities.
2. if i send a message to C, their LED lights up in green. But B gets their LED lighted up in red.

Lesson plan table:

<https://docs.google.com/spreadsheets/d/1kOG-Smnd3Jc069m-o15BCuXtxpls5Lut8Y6f8QgHDl8/edit?gid=0#gid=0>

Lexicon

Group 1: 9th and 10th Grades

Group 2: 11th and 12th Grades

Challenge cards

**Challenge Cards:**

**How would you execute these?**

1. Visual Cryptography: Go through what visual cryptography is <https://www.101computing.net/visual-cryptography/> and use the technique to encrypt a message you want to share with others.   
     
   The encrypted message is then asked to be decrypted by the other students in their own time.

**Relevance:** Class 4 (when encryption is taught)

1. **Digital footprint:** Do a backtracking and identify where all you left digital footprints.   
     
   **Activity**: Wikipedia internal links. Two students are given a random Wikipedia page irrelevant to a theme. They are supposed to click on the internal links on the page and reach the final page relevant to the theme.   
     
   Think about how a digital footprint works. You search something on amazon and the ad for the same product appears on instagram or youtube, how?   
     
   *Will have to give an example here for how they can track back their digital footprint*
2. **Terms and Conditions**: For your favourite or most-used website/app, read the T&C and Privacy Policies. Identify what all data they are collecting about you, why they are interested in the data, and whether you should care about it.   
     
   **Relevance:** Class 1 (when they are learning about cookies, digital tracking, etc.)  
     
   [Home Page - ToS;DR](https://tosdr.org/en) Terms of service; Didn’t read : provides info on the tracking habits of different websites in short. Can give this to students who successfully solve this challenge card.
3. **Invariants:** Understand what invariants are and identify the invariants of the very basic algorithms we are writing in the class.  
     
   **Relevance:** Class 11 and 12.
4. Read about public key and private key, and understand how they are used in end-to-end encryption  
     
   **Relevance:** Before class 7 so that they can use these ideas for their final mesh project.
5. What could they do to avoid sending a message to someone in between? Say, A sending a message to C means B lighting up first. So, what can A do to prevent B from lighting up and directly sending a message to C?  
     
   **Relevance:** Class 6 when they are messing around with the network and sending messages across to each other.

More challenging challenge cards:  
  
1.

Pt.1 **Nested loops - medium**

1

1 2

1 2 3

1 2 3 4

Try to write a code that prints out the above pattern. Think of how having a loop inside another loop can achieve this.

Pt. 2 **The cult of pattern - hard**

You have just found a coded ritual pattern in an old manuscript. If you print it exactly for n = 5, it reveals the hidden symbol of an ancient cult.

The pattern looks like:

\* \*

\* \*

\* \*

\* \*

\*

\* \*

\* \*

\* \*

\* \*

**Challenge:**

1. Recreate this using loops.
2. Then, refactor your code into 2 or more functions — because the pattern is *symmetrical*. Can you figure out the best way to break it up?

**2. Are they soulmates? - medium**

You are given names of two people (your function inputs). Your task is to determine if their **relationship** is “magical.” A magical bond exists if the string formed by combining both names (ignoring spaces and cases- account for these using appropriate python functions) reads the same forwards and backwards.

**Example:** Input: "Anna" and "N a n a" → Output: “Magical!”  
 ("annanana" reversed is the same)

3. **Understanding Caesar ciphers** - Easy

You’ve been hired to help a robot design a welcome banner using only letters and ASCII codes.

Your task is to write a function that:

1. Takes a word as input (e.g., "HELLO").
2. Converts each letter into its **ASCII value** using ord().
3. Then, **shifts** each ASCII value by +1 and prints the new character using chr().

**Input: "DOG"**

**Output: "EPH" # 'D' → 68 → 69 → 'E', and so on**

**Extension:** Can you recover the initial message?

5. **Digital Biometrics - easy**

What is biometric data? What forms of it do you use every day without realizing?

Where is your data going — stored locally on device, or uploaded to a server?

What’s the biggest risk if biometric data is leaked? (Hint: you can change your password, but not your face.)

Look into one real-world incident (e.g., Aadhaar biometric leak, Clearview AI, etc.) and write a 2–3 line summary of what happened.

6. **2-D arrays introduction - easy**

Your classroom has 3 rows and 4 columns of seats. The seating arrangement is stored in a 2-D list:

seats = [

["Alice", "Bob", "Carol", "David"],

["Eve", "Frank", "Grace", "Heidi"],

["Ivan", "Judy", "Mallory", "Niaj"]

]

1. Print the name of the student sitting in the **2nd row, 3rd column**
2. Print all students in the **first row**
3. Print the **entire chart**, one row per line
4. Ask the user for a name and tell whether the student is **present** or **absent**

Script

1. set execution (part of the command not in the script for windows)
2. check if the hardware is plugged in
3. check for the internet (ashoka wifi connected), ping 1.1.1.1
4. check for the relevant package manager
   1. winget for windows
   2. homebrew for mac  
       xcode-select  
       run the script as sudo on mac
   3. apt for ubuntu/debian  
       build-essentials  
       we might have to look for packages on snap  
      *get deepraj’s script*
5. install the packages
   1. python 3.12
   2. vscode   
      install the vscode extensions (python, pylance, ipython, serial monitor)
   3. git
6. clone the repo (git clone <desktop path>)
7. set up the virtual env
8. install the dependencies of the project
9. code <path> (call vscode to the right folder)
10. make the curl request

**note**:

1. have checks for when they use the incorrect script
2. dont install vscode as sudo but others yes.
3. optional: script to do a cold run of who all have done - to display a leaderboard on the screen. hardware ID if that is plugged in.
4. think about gamification
5. if they already have the VScode installed then let them manually follow the process. Those who are already comfortable with all these, let them handle it themselves.
6. verbose checkpoints on the terminal. colored logs.
7. last day: cleanup script that reverses all these – not mandatory. instead of uninstalling but set execution policy to default and delete the virtual env folder.

make two scripts:

one for the installation of vscode and its packages (python, jupyter) on the 1st day and the other for setting up the network on the 5th day

**Guide For Students:**

Setting up the system:

Coding Basics

#### Quick Refresher

#### 1. Syntax & Structure

* Python blocks are defined by indentation (PEP 8 recommends 4 spaces).
* A statement ends at the newline unless a backslash \ or open bracket continues it.

#### 2. Data Types & Variables

* Built-ins: int, float, complex, bool, str.
* Use type(x) to inspect; cast with int(), float(), complex(), str().

#### 3. Functions

**def** **greet**(name: **str**) -> **str**: # type hints optional

**return** f"Hello {name}"

Copy

Everything not return-ed is discarded—no implicit printing here.

#### 4. Control Flow (flat only)

**if** x > 0:

sign = "positive"

**elif** x == 0: # one elif chain allowed

sign = "zero"

**else**:

sign = "negative"

**while** n > 0: # single-level loop

total += n

n -= 1

Copy

#### 5. Operators

Arithmetic + - \* / // % \*\*, comparisons, and logical and or not can all be used directly inside expressions.

#### 6. Basic Error Handling

**try**:

result = a / b

**except** ZeroDivisionError:

**raise**

**else**:

log(result)

**finally**:

cleanup()

Copy

Ready? Jump to the activities!

#### Syntax & Comments

1

**Spot the Error**

Which option runs without a syntax error?

**def** **say\_hi**()

**return** "Hi"

Copy

x = 5

**if** x > 3

y = 1

Copy

x = 5

**if** x > 3:

y = 1

Copy

Submit

Start project before checking your activities.

2

**Choose the Comment**

Select the valid multi-line comment style.

/\* multi \*/

// comment

"""multi-line\ncomment"""

Submit

Start project before checking your activities.

#### Data Types & Variables

3

**Complete add\_two\_numbers(a, b) function**

Write a function that returns the arithmetic sum of a and b.

Check Activity

Start project before checking your activities.

4

**Extract the Last Character from a String**

Create a function named last\_char that returns the last character of a non-empty string text. This demonstrates string indexing in Python.

Check Activity

Start project before checking your activities.

5

**Identify the Data Type**

What data type does Python assign to the expression 3 + 4j?

int

complex

float

Submit

Start project before checking your activities.

6

**Convert String Booleans to Actual Boolean Values**

Create a function named to\_boolean that converts string representations of booleans ("True"/"False", any case) to actual Python boolean values. Raise ValueError for invalid inputs. This demonstrates type conversion and error handling.

Check Activity

Start project before checking your activities.

#### Functions & Scope

7

**Convert Fahrenheit to Celsius with Rounding**

Create a function named fahrenheit\_to\_celsius that converts Fahrenheit temperatures to Celsius and rounds the result to 1 decimal place. This demonstrates arithmetic operations and the round() function.

To convert Fahrenheit (°F) to Celsius (°C), use the following formula:

‎

°

C

=

5

9

×

(

°

F

−

32

)

°C=59×(°F−32)

‎

Check Activity

Start project before checking your activities.

8

**Understand Python Variable Scope Rules**

Which code does not raise UnboundLocalError ? Select the code that correctly handles variable scope.

x = 1

**def** **foo**():

x += 1

**return** x

Copy

**def** **foo**():

y += 2

**return** y

Copy

x = 1

**def** **foo**():

**return** x + 1

Copy

Submit

Start project before checking your activities.

#### Control Flow

9

**Check if a Number is Even**

Create a function named is\_even that determines whether a given number is even, returning True or False. This demonstrates the modulo operator and boolean logic.

Check Activity

Start project before checking your activities.

10

**Determine the Sign of a Number**

Create a function named sign\_of\_number that returns "positive", "zero", or "negative" based on the sign of an integer. This demonstrates conditional branching with if-elif-else.

Check Activity

Start project before checking your activities.

11

**Calculate Sum of Numbers from 1 to n**

Create a function named sum\_until that calculates the sum of all integers from 1 to a given number n (inclusive). Raise a ValueError for negative inputs. This demonstrates loops and error handling.

Expected outcome:

* For n = 5, the function should return 15.
* For n = 1, the function should return 1.
* For n = -1, the function should raise a ValueError with the message "n must be non-negative".

Check Activity

Start project before checking your activities.

Readings

**Class Resources**

**Day 1**

1. [What cookies are and how they work!](https://www.youtube.com/watch?v=s04Vjlcgwco)
2. [How ads follow you around the internet](https://www.youtube.com/watch?v=HFyaW50GFOs)

**Day 2**

1. [The truth behind filter bubbles: Bursting some myths | Reuters Institute for the Study of Journalism](https://reutersinstitute.politics.ox.ac.uk/news/truth-behind-filter-bubbles-bursting-some-myths)

**Day 3**

1. Filter bubbles: <https://www.youtube.com/watch?v=prx9bxzns3g&t=15s>
2. Data brokers: [Data Brokers: Last Week Tonight with John Oliver (HBO) - YouTube](https://www.youtube.com/watch?v=wqn3gR1WTcA)
3. Cambridge Analytica: [How Cambridge Analytica Exploited the Facebook Data of Millions | NYT](https://www.youtube.com/watch?v=mrnXv-g4yKU)
4. Censorship: <https://www.khanacademy.org/partner-content/wi-phi/xd226e27a:emerging-technologies/xd226e27a:censoring-hate-speech/v/philosophy-emerging-technologies-3-should-online-platforms-censor-hate-speech>
5. In Defence of Anonymity: <https://www.youtube.com/watch?v=1ZnXPVyWP8w>
6. Cyberethics: <http://irep.iium.edu.my/23680/>

**Day 4**

1. History of encryption: [11 Cryptographic Methods That Marked History: From the Caesar Cipher to Enigma Code and Beyond](https://interestingengineering.com/innovation/11-cryptographic-methods-that-marked-history-from-the-caesar-cipher-to-enigma-code-and-beyond)
2. Understanding how QR codes work: [Anatomy of a QR Code: Know Its Structure & Functionality](https://www.qrcode-tiger.com/anatomy-of-a-qr-code)
3. BAse64 encoding: [Base64 Encoding](https://www.youtube.com/watch?v=aUdKd0IFl34)
4. Letter frequency: [Frequency Table](https://pi.math.cornell.edu/~mec/2003-2004/cryptography/subs/frequencies.html)
5. To show in class: [Diffie-Hellman Key Exchange](https://www.youtube.com/watch?v=YEBfamv-_do) (2.47 onwards)
6. RSA explainer: <https://www.youtube.com/watch?v=hm8s6FAc4pg&t=224s>

**Python Resources:**

1. [Python Cheat Sheet & Quick Reference](https://quickref.me/python.html) - For those that have coded in other languages before and need a syntax reference for Python.
2. <https://futurecoder.io/>
3. <https://www.freecodecamp.org/news/learning-python-from-zero-to-hero-120ea540b567/>
4. <https://www.youtube.com/watch?v=qwAFL1597eM>
5. <https://www.youtube.com/watch?v=eWRfhZUzrAc>
6. 50 Days of Python by Benjamin Bennett Alexander
7. Leetcode
8. Loops: <https://www.youtube.com/watch?v=2Icpbawb-vw>

Tab 14