**CODING CLUB**

**PROPOSAL FOR COMPETITIVE CODING INITIATIVE**

**BRIEF**

We the students of MIET Jammu of 5th semester want to make a humble request of taking an initiative of leading COMPETITIVE CODING for 1st and 3rd semester. The coding club would solely be directed to project based learning and visualisations. We hope to achieve a high quality coding environment among the students of the MIET and prepare the students for the market by enhancing their problem solving abilities and the ability to understand a problem in a much better way so as they come with much more efficient solutions to the problems. We hope to leave a legacy of high performance coders who will motivate the upcoming generation of students. Our goal will solely lead to generate a high performance coders.

**MOTIVATION**

It is always said to be the change you want to see in society. We are an integral part of the future of our country and want all around us to flourish and prosper in the best possible way.

As we know India alone produces more computer science engineer than most of the northern and south eastern countries combined. In this situation it becomes almost impossible to compete and exist as a established entity. Almost 85% of indian engineers are either unemployed or are not fit for any job at all. This is the cause of the failed western culture that has been integrated to the Indian college education system that keeps producing employees with a bottlenecked mindset and the need for an external motivation. It brings the members of this club immense disgust to see the stats and the consequences that has shaped the engineering environment in india in such a terrible form. People in India Need to realise that what people are taught and what is required by the market is completely different .

We the members of the CODING CLUB group do not want to support or promote this environment of failed society that is not even able to prove its worth.

Before the situation becomes even tougher and Students of MIET end up unemployed or good for nothing an initiative needed to be taken for a coding club that not only improve the intellect of the student but also developes an ability of problem solving .

**OBJECTIVE**

The main objective of the coding club is to develop and uplift an environment of competitiveness and high intellect.

Our objective will revolve around the approach to solve the real world problems irrespective of the language we choose to solve the problem in.

We as being the seniors for the 1st semster student want the 1st semester students to be influenced and motivated by such a positive environment that enables them to think and act in an innovative way.

We want to develop a skill of solving the problems in the minds

before even writing the code among all the students. This way the students will be able to completely understand a given problem and workout multiple approaches to the solution.

To teach is to learn. So, while conducting the sessions, the group members will also learn along and add students to our group- both seniors and juniors who are extremely excited to solve real world problems and gain some exceptional skills.

**IMPACT**

# **•** The coding club will play a great role in enhancing the overall competitive environment among the students of all the semesters .

# **•** It will play a great role in building a sense of problem solving among the students in a very young age .

# **•** The coding club will provide all the necessary tools and hardwares that are required to solve a particular real world problem so the students get all the top notch resources and they get no reason to not being able to code and implement a machine learning model .

**CHALLENGES**

1. Internet issues in the college premises as only 50 users are able to connect to a single modem.
2. Not everyone has the same capability and interest towards coding and solving real world problems .
3. The students might picture the coding club sessions as an addition to the curriculum and might skip the classes so as in order to not build up the burden and feel pressurised .

**APPROACH**

We will create groups of people who have expertise in particular domain and will deploy them at certain time slots to teach and enlighten the students. Our approcah is completely going to be the Visualisation and project based learning. The students will be made to practice and implement solutions as well so that they try to understand the problem completely before writing the code. We will conduct small mooc test , MCQ problem statements and try to engage as much students as we can .

We will introduce competitive platforms like Code Chef , Leet Code , HackerRank etc so as the students aslo get to showcase the level of problems that the students are able to solve. We will also encourage the students to develop their LINKEDIN and GITHUB profiles. So the recruiter would get an amazing impression the moment they take a look at the profiles and résumés.

**KEY OUTCOMES**

**•** Improved quality of conversation among the young students.

**•** Better chances of employment through top notch companies.

**•** Increased involvement of students in coding competitions, hackathons, seminars and technical events .

**•** Better grasp on multiple languages to solve a particular problem in different ways.

**•** Developing a sense of competitivenes among the students so as to grow and improve as a group.

**•** Leaving a legacy of high performance coders and problem solvers that shape the world the way it needs to be.

**WORKFLOW**

**•** We will start with the basics of programmig languages irrespective of the programming language and move our way ahead to the complex programming , doing the project work as well while we progress .

**•** We will introduce new age technologies and ways to write and run programs so as to make the solution more efficient and fast.

We will try and engage more student to the mooc tests and MCQ interactive sessions so the students learn and grow together as a group .

**•** Then we will work towards the competitiveness among the students , by introducing the competitive platforms like Code Chef , Hacker Rank , Leet code , Algoexpert.io etc.

**•** After this we will work on the professional profiles of the student a the platforms like the GITHUB , LINKEDIN etc. by giving them proper resources and training so that their profiles get highlighted and short listed the moment someone lays eyes on the profile .

**•** This will be followed by encouraging deserving students to join our coding club to continue the legacy for high performance coders that we leave behind.

**TEAM STRUCTURE**

**Disclaimer : The Team members listed in the below list are not reserved for their respective positions . Any member at any point of their service may be replaced or introduced as per the requirement and approval of the coding club committee.**

1. **CORE COMMETTE :**

Resource Approval / Execution Plan

**• Archit Prasar (5th Sem A1 Cse dept)**

**• Samragyi Vats (5th Sem A1 Cse dept)**

1. **RESOURCE TEAM :**

Development

**• Archit Prasar (5th Sem A1 Cse dept)**

**• Lavanya Sharma (5th Sem A1 Cse dept)**

Artificial Intelligence

**• Ishav Verma (5th Sem A Cse dept)**

Cybersecurity

**• Aadhaar Koul (5th Sem A1 Cse dept)**

Design

**• Lavanya Sharma (5th Sem A1 Cse dept)**

Blockchain

Python

**• Archit Prasar (5th Sem A1 Cse dept)**

Web Development

**• Archit Prasar (5th Sem A1 Cse dept)**

C , C ++

**• Farheen Shaad (3th Sem A1 Cse dept)**

Internet of things

Data structures and algorithmn

1. **HUMAN RESOURCE TEAM**

EScreening

**• Ishita Sharma (5th Sem A1 Cse dept)**

Recruiting

**• Ishita Sharma (5th Sem A1 Cse dept)**

Downsizing

**• Ishita Sharma (5th Sem A1 Cse dept)**

1. **MANAGEMENT TEAM**

Social media

**• Samragyi Vats (5th Sem A1 Cse dept)**

Documentation

**• Samragyi Vats (5th Sem A1 Cse dept)**

**• Farheen Shaad (3th Sem A1 Cse dept)**

**• Aadhaar koul (5th Sem A1 Cse dept)**

Blogs

**• Samragyi Vats (5th Sem A1 Cse dept)**

Forums

**• Samragyi Vats (5th Sem A1 Cse dept)**

COURSE CURRICULUMS

**CURRICULUM FOR PYTHON PROGRAMING**

Overview

This curriculum will take the learners through the concepts of python programming right from

the basics to intermediate and then advanced. Later “Creative Projects” will be assigned

that can be completed in small groups or individually. Working in small groups is sometimes

required by state standards. These projects can be scaled up or down to meet available

class times.

The course curriculum can be defined as following topics:

● Fundamentals of Python

• Introduction to Python

• Running Python Programs

• Writing Python Code

● Working with Data

• Data Types and Variables

• Using Numeric Variables

• Using String Variables

● Input and Output

• Printing with Parameters

• Getting Input from a User

• String Formatting

● Making Decisions

• Logical Expressions

• The “if” Statement

• Logical Operators

• More Complex Expressions

● Finding and Fixing Problems

• Types of Errors • Troubleshooting Tools

• Using the Python Debugger● Lists and Loops

• Lists and Tuples

• List Functions

• “For” Loops • “While” Loops

● Numeric and Date Functions

• Dates and Times

• Advanced Data and Time Management

• Random Numbers • The Math Library

● Working with Strings

• Character Data

• String Functions

• Input Validation with “try / except”

● Functions

• Writing and Calling Functions

• Function Inputs and Outputs

• Local and Global Scope

● Python Classes

• Thinking about Objects

• Class Variables and Methods

• Managing Class Files

● Class Instances

• Creating Objects with Instance Data

• Instance Methods

• Managing Objects

● Creative Project

• Project Life-cycles and teams

**CURRICULUM FOR C PROGRAMING**

**Overview**

This curriculum will take the learners through the concepts of C programming right

from the basics to intermediate and then advanced. To thoroughly implement this ,

the curriculum is divided into eight parts.

They are as follows :-

**1. Introduction to C language language**

• C language introduction

• Features of C language

• Benefits of C over other languages

• Compilation of C programs

• Hello world program in C

2. **Going through Variables, Data Types & Operators**

• Variables and Keywords in C

• Data Types in C

• Operators and its types

**3. Understanding Control Flow statements**

• Loops in C

• Decision making statements in C

• Switch statement in C

• Continue Statement | Break Statement

• Loops and Control statement practice questions

**4. Array and String Handling in C**

• Arrays in C

• Strings in C

• String functions in C

• Single - quoted and Double - quoted declaration of the char array**5. Familiarisation with Functions**

• Functions in C

• Function Prototypes

• Recursion concept

**6. Learning about Pointers , structures and unions**

• Pointers in C | Double pointer

• Structures | Union | Enumeration in C

• Declare a pointer to a function

• Pointer vs Array in C

• Operations on struct variables in C

**7. Understanding Dynamic Memory Allocation (DMA)**

• Dynamic Memory Allocation in C

• Difference between malloc and calloc

**8. File management in C**

• File handling in C

• Merging Files

• File Operations

**Course Outcomes :**

• Learners will be able to implement and showcase their skills in C programming.

• Learners will be able to design modular programs in C language .

**Curriculum for Artificial Intelligence**.

Under this curriculum the learners would be taught about the Artificial Intelligence & it’s real-life

applications. For this the curriculum of the course is divided into four parts with whole duration of

24 Hours. They are as:

1. Python Basics. (6 Hours)

2. Maths Basics. (2 Hours)

3. Artificial Intelligence. (12 Hours)

4. Useful Platforms for AI Development. (4 Hours)

The course curriculum can be defined as following topics:

1. **Python Basics**:

a. About Python:

i. What is Python?

ii. Features of Python.

iii. Uses of Python.

iv. Environment Setup.

v. IDE Setup: VS Code, Jupyter Lab, Google Colab & IBM Watson Studio.

b. Python Basic concepts:

i. Programming Basics in Python.

ii. Python Data Types.

iii. Control Flow Statements.

iv. Functions.

v. Objects & Classes.

c. Python Advanced concepts:

i. Python regular Expressions.

ii. I/O Operations.

iii. Exception.

iv. Data Structures.

d. Third Party Libraries.

i. Introduction to ‘pip’.

ii. Useful ‘pip’ commands.

iii. Numpy.

iv. Pandas.

v. Matplotlib.

vi. Scipy.

vii. Scikit-Learn.

viii. Tensorflow – Tensorflow, tensorflow-gpu, keras, CUDA & cuDNN.

e. CUDA & cuDNN setup with Nvidia GPU(s).

2. **Maths Basics**:

a. Relation of Mathematics & AI.

b. Linear Algebra:

i. Basic Concepts & Operations.

ii. Linear Transformation & Matrix iii. Matrix Decomposition.

c. Probability & Statistics:

i. Basics of Probability & Statistics.

ii. Random variable & probability distribution.

iii. Estimation.

d. Optimization Problem:

i. Classification of Optimization problem.

ii. Gradient descent method.

3. **Artificial Intelligence**:

a. Artificial Intelligence:

i. What is AI?

ii. History of AI.

iii. Future scope of AI.

iv. How to achieve AI.

v. Introduction to AI, ML & DL.

vi. Difference among AI, ML & DL.

b. Data Cleaning:

i. What is Data & Datasets?

ii. How to get Datasets?

iii. Need of Data.

iv. Data Examination.

v. Data Cleansing.

vi. Understanding features & labels

vii. Process of feature selection.

c. Machine Learning:

i. What is ML?

ii. Types of ML.

iii. Different types of ML algorithms.

iv. Process of ML.

v. Case study & Hands on Practice.

d. Deep Learning:

i. What is DL?

ii. DL Algorithms.

iii. Activation Function, Normalizer & Optimizer.

iv. What are neural Networks?

v. Types of Neural Networks.

vi. Case study & Hands on Practice.

e. Model Building:

i. Model Selection.

ii. Model Building.

iii. Model Exporting.

iv. Model Deployment.

v. Model Management

**4. Useful Platforms for AI Development:**

a. Github:

i. What is Github?

ii. What is Git & Gitbash?

iii. Github Codespaces.

iv. Git commands.

b. Kaggle:

i. What is Kaggle?

ii. Kaggle Competitions.

iii. Finding Datasets on Kaggle.

c. IBM Watson:

i. What is Watson?

ii. Watson Studio.

iii. Watson APIs.

d. NVIDIA:

i. What is NVIDIA?

ii. NVIDIA AI.

iii. NVIDIA Omniverse.

iv. NVIDIA GPU(s).

v. NVIDIA CUDA & cuDNN.

e. UiPath:

i. What is RPA?

ii. How to use RPA?

iii. RPA & AI.

iv. Integrating AI using RPA.

**Outcomes of Course:**

 Understanding how python can be used for AI Development.

 Understanding various ML Algorithms & their implementation.

 Understanding about Neural Networks & their implementation.

 Understanding the use of AI to solve Real-life Problems.

 Build & deploy own projects.

**CIRRICULUM FOR CYBERSECURITY**

In this course learners will be taught ethical hacking from

scratch . No prior knowledge is required to pursue the

curriculum . At the end of the course learners would be able to

identify flaws , break things down to build better solutions.

1. Introduction:

\* Course Overview

\* What , Why , when , where and hows of cybersecurity

2. Setting up a hacking lab:

\* Initial Prepration

\* Installing Kali , Tails and Ubuntu as VM on windows

3. Linux Basics:

\* Basic Overview

\* The Linux Filesystem

\*Understanding sudo and sudo su

\* The Terminal and linux commands

\*Finding Files in kali

\*Managing Kali Linux Services - SSH , HTTP , FTP

\* Installing and removing tools - apt , pip , apt-get , dpkg etc

4. Command Line Fun :

\*The bash Environment

\*Bash Scripting

\* Text Searching and Manipulation : grep , sed , cut , awk ,

nano , vi , comm , diff , vimdiff , etc

5. Network Hacking:

\*Introduction to penetration Testing

\*Networking Basics

\* Connecting a wireless adapter to kali\* What is mac how to change it ?

\*Wireless Modes (Managed and Monitored)

\*Wireshark

\*tcpdump

6. Hardware for hacking:

\*Raspberry pi

\*Arduino

\*Rubber Ducky , Scripting

\* Wifi Pineapple

\*HackRF

\*OMG Cable

\*IOT for Hacking

7. Social Engineering :

\* hacking an Organisation

\* Tackticks For Hacking

\* Power of listenig

\* Demonstrations

8. Information Gathering :

\* Taking Notes

\* Website reach

\* Google hacking

\*Netcraft

\* Shodan

\*Pastebin

\*Email Harvesting

\*Password Dumping

\*OSINT Framework

\*Malteago

\*DNS Enumeration

\*Port Scanning

\*SMB Enumeration

\*SMTP Enumeration9. Network Hacking :

\*Packet Sniffing Basics

\*Wifi Bands - 2.4Ghz and 5Ghz

\*Targetted Packet Sniffing

\*Deauthentication Attack (Disconnecting Any Device from

the network)

10. Network Hacking :

\*Gaining Access introduction

\*Theory Behind Cracking WEP Encryption

\*WEP-Cracking Basics

\*Fake Authentication Attack

\*ARP Request Replay Attack

11. Network Hacking - Gaining Access - WPA/WPA2 Cracking :

\* Intro to WPA and WPA2 Cracking

\*Capturing the Handshake

\*Creating the wordlist

\*Cracking WPA and WPA2 Using a wordlist Attack

12. Network Hacking - Post Connection Attacks - Information

Gathering :

\*Installing Windows as a VM

\*Discovering Devices Connected to the same network

\*Gathering sensitive info about connected devices

\*Gathering More Sensitive info

13. Network Hacking - Detection and security :

\*Detecting ARP Poisoning Attacks

\* Detecting Suspicious Activities in the Network

\* Preventing MITM Attacks - Methods

14. Website Hacking :

\* Burpsuite

\*SQL injection Attacks

\* Client Side attacks\* Server side attacks

\*Information Gathering

\*Discovering and reporting Vulnerabilities

15. Vulnerability Scanning :

\*Overview

\*Scanning With Nessus

\*Scanning With Nmap

16. Reporting :

\* Documentation

\*Vulnerability Assessment

\*Pentesting Reports

\*Reporting and legal terminology

1. Cybersecurity Certifications:

\*eJPT

\* NDE

\* EHE

\* DFE

\* CEH

\* CCNA

\* CCIE

\* OSCP

\* OSCE

Curriculum for Web Development

Curriculum Overview :

This curriculum will take the learners through the concepts of web development right from

the basics to intermediate and then advanced. Later “Creative Projects” will be assigned

that can be completed in small groups or individually. Working in small groups is sometimes

required by state standards. These projects can be scaled up or down to meet available

class times.

The course curriculum is divided into two parts:

Front-End development

Back-End development

Front-End development:

● Introduction to Front-End Engineer Career Path

Welcome to the Front-End Engineer Career Path!

● Overview of Web Development

Be introduced to the field of front-end web development.

● Fundamentals of HTML

Learn to build a structure for a website using HTML.

● Fundamentals of CSS

Learn how to apply styles to HTML documents using CSS.● Developing Websites Locally

Get acquainted with text editors and Chrome DevTools to develop

websites locally.

● Deploying Websites

Learn how to deploy websites using GitHub Pages and the command

line.

● Improved Styling with CSS

Explore intermediate topics in CSS to customize styles and

implement navigation elements.

● Fundamentals of Web Design

Explore the best user interface (UI) design practices to implement

CSS.

● Making a Website Responsive

Learn how to utilize responsive web design practices using CSS grid,

flexbox, and media queries.

● JavaScript Syntax, Part I

Use JavaScript to create basic programs that can store and

manipulate various types of data.

● JavaScript Syntax, Part II

Explore JavaScript to create complex programs using arrays, loops,

objects, and iterators.

● Building Interactive Websites

Implement JavaScript to add interactive experiences to a website.● Making A Website Accessible

Get acquainted with the best accessibility (a11y) practices to ensure

that your websites are usable to everyone.

● CSS Transitions and Animation

Learn how to create visually dynamic websites using CSS transitions

and animations.

● Git and GitHub, Part I

Explore Git and GitHub to version control your programs.

● Portfolio Project: Personal Portfolio Website

Bring together what you have learned in the previous lessons and

build a project off of Codecademy.

● JavaScript Syntax, Part III

Level up your JavaScript by learning how to implement classes,

modules, and error handling.

● JavaScript Testing

Using Mocha and Chai, learn various JavaScript testing

methodologies, such as Test-Driven Development (TDD).

● Async JavaScript and HTTP Requests

Learn about APIs (Application Programming Interfaces). Working

with APIs will enable you to work with data stored on remote servers.

● Web Apps

Be introduced to web applications and learn about single-page

applications (SPAs) and how they are different from static websites.Back-End development :

● Welcome to the Back-End Engineer Path

The first steps to your new career as a back-end software engineer.

● Setting Up a Dev Environment

Learn about software engineering and the tools developers use to

create websites.

● Web Development Fundamentals

Learn about web development and create your first web page on the

internet.

● JavaScript Syntax, Part I

Use JavaScript to create programs that can store and manipulate

data.

● JavaScript Syntax, Part II

Create more complex programs using arrays, loops, objects, and

iterators.

● Building Interactive Websites with JavaScript

Implement JavaScript to add interactive experiences to a website.● JavaScript Syntax, Part III

Learn intermediate JavaScript by implementing classes, modules,

and error handling.

● Git and Github, Part I

Explore Git and GitHub to version control your programs.

● Portfolio Project: Mixed Messages

Build a Node.js console app that generates random messages each

time a user runs the program and version of your project with

Git/GitHub.

● Basics of Back-End Development

Learn about back-end development and programming servers.

● Async JavaScript and HTTP Requests

Learn about APIs (Application Programming Interfaces). Working

with APIs will enable you to work with data stored on remote servers.

● Build a Back-End with Express.js

Learn how to create back-end servers and APIs in JavaScript using

the popular Express.js framework.

● Git and Github, Part II

Learn how to use Git and GitHub to collaborate efficiently with

developers.

● Portfolio Project: Budget I

Use Node and Express.js to build an API that allows users to create

and manage a personal budget.● Database Basics

Get acquainted with databases and how they help store, retrieve, and

manipulate data.

● Working with Databases

Craft more sophisticated SQL queries and calculations to build

data-intensive applications.

● Designing Relational Databases

Learn how to design relational databases that you can then

implement in PostgreSQL.

● Advanced PostgreSQL

Learn about database performance and techniques for efficiently

accessing data and maintaining optimal performance.

● Connecting a Database to a Server

Learn how to integrate a PostgreSQL database into your applications.

● Deploying a Server

Learn how to build an API with Node, Express, and PostgreSQL and

deploy it to Heroku.

● JavaScript Testing

Understand the fundamentals of Test Driven Development (TDD) and

how you can apply its principles when developing back-ends.

● Software Design Principles

Start designing more complex systems with techniques such as

design patterns, SOLID principles, and UML.● API Development with Swagger and OpenAPI

Learn how to develop APIs using Swagger tooling and the OpenAPI

specification.

● Portfolio Project: Budget II

Take your budget app one step further and add a new layer to the

project by setting up a database.

● Web Security Fundamentals

Learn intermediate concepts in web security including prevention

techniques for common threats.

● User Authentication and Authorization

Learn how to protect your resources with authorization and

authentication techniques.

● Data Security

Learn to secure your data using Transport Layer Security, Role-Based

Access Control, and more!

● Common Attacks on Web Applications

Explore common threats that web applications face and how to

mitigate them.

● Fundamentals of Operating Systems

Learn about operating systems by deep diving into each of their main

functionalities.

● Caching and CDNs

Learn about how content is stored on different servers with caching

and CDNs.● Scalability

Learn different strategies for scaling a software system.

● DevOps Fundamentals

Learn the fundamentals of DevOps, a culture supported by practices

and tools