**Lecture Title: Modern Programming Approaches: Leveraging Large Language Models (LLMs) for Coding**

**Introduction:**  
Good morning, everyone! Today, we’re going to explore a fascinating and modern approach to programming that is transforming the way people write code. Even if you’re not a computer science major, this lecture will show you how tools like Large Language Models (LLMs) can make coding more accessible, intuitive, and efficient. By the end of this session, you’ll understand how LLMs like ChatGPT, GitHub Copilot, and others are changing the game for programmers and non-programmers alike.

**1. What is a Large Language Model (LLM)?**

Before we dive into programming, let’s start with the basics. A Large Language Model (LLM) is an artificial intelligence system trained on vast amounts of text data. It can understand and generate human-like text, answer questions, and even write code. Examples include OpenAI’s ChatGPT, Google’s Gemini, and GitHub’s Copilot.

Think of an LLM as a super-smart assistant that can help you with tasks like writing essays, solving math problems, or—most importantly for us today—writing code.

**2. Why Use LLMs for Coding?**

Coding can seem intimidating, especially if you’re not a computer science major. But LLMs are here to help! Here’s why they’re so useful:

* **Accessibility:** You don’t need to memorize complex syntax or programming rules. LLMs can generate code for you based on simple instructions.
* **Efficiency:** LLMs can write code faster than humans, saving you time and effort.
* **Learning Tool:** LLMs can explain code in plain language, making it easier to understand how programming works.
* **Creativity Boost:** LLMs can suggest new ideas or approaches to solving problems.

**3. How Do LLMs Help with Programming?**

Let’s break down how you can use LLMs for coding:

**a. Writing Code from Scratch**

You can describe what you want the program to do in plain English, and the LLM will generate the code for you. For example:

* **You say:** “Write a Python program that calculates the average of three numbers.”
* **LLM writes:**

python

Copy

num1 = float(input("Enter first number: "))

num2 = float(input("Enter second number: "))

num3 = float(input("Enter third number: "))

average = (num1 + num2 + num3) / 3

print("The average is:", average)

**b. Debugging and Fixing Errors**

If your code isn’t working, you can paste it into an LLM and ask for help. The LLM will identify the error and suggest fixes.

**c. Explaining Code**

If you come across code you don’t understand, you can ask the LLM to explain it line by line.

**d. Automating Repetitive Tasks**

LLMs can help you write scripts to automate boring tasks, like renaming files or organizing data.

**4. Real-World Examples of LLMs in Action**

Let’s look at some practical examples of how LLMs are used in programming:

**a. GitHub Copilot**

GitHub Copilot is an AI-powered coding assistant that integrates with code editors like Visual Studio Code. It suggests code snippets as you type, almost like autocomplete for programming.

**b. ChatGPT for Prototyping**

You can use ChatGPT to quickly prototype ideas. For example, if you’re building a website, you can ask ChatGPT to generate HTML, CSS, and JavaScript code for a basic layout.

**c. Data Analysis**

LLMs can help you analyze data by writing Python scripts for libraries like Pandas or Matplotlib. For instance, you can ask for a script to create a bar chart from a dataset.

**5. Benefits for Non-Computer Science Majors**

If you’re not a computer science major, LLMs can still be incredibly useful:

* **Problem Solving:** You can use LLMs to write small programs that solve specific problems in your field, like analyzing data for a biology project or automating calculations for an engineering assignment.
* **Learning by Doing:** Instead of spending hours learning programming syntax, you can focus on solving real-world problems with the help of LLMs.
* **Collaboration:** LLMs can help bridge the gap between technical and non-technical team members by translating ideas into code.

**6. Limitations and Ethical Considerations**

While LLMs are powerful, they’re not perfect. Here are some things to keep in mind:

* **Accuracy:** LLMs can sometimes generate incorrect or inefficient code. Always test and review the code they produce.
* **Over-Reliance:** Don’t rely solely on LLMs. It’s important to understand the basics of programming so you can troubleshoot and improve the code.
* **Ethics:** Be mindful of using LLMs responsibly. Avoid using them to plagiarize or create malicious software.

**7. Getting Started with LLMs for Coding**

Here’s how you can start using LLMs for programming:

1. **Choose a Tool:** Try tools like ChatGPT, GitHub Copilot, or Google Gemini.
2. **Start Small:** Begin with simple tasks, like writing a calculator program or automating a repetitive task.
3. **Experiment:** Play around with different prompts and see how the LLM responds.
4. **Learn Along the Way:** Use the LLM’s explanations to improve your understanding of programming concepts.

**8. Conclusion**

In today’s world, programming is no longer just for computer science experts. With the help of LLMs, anyone can write code, automate tasks, and solve problems. Whether you’re a biologist, an engineer, or a business student, these tools can empower you to bring your ideas to life.

Remember, LLMs are not here to replace programmers—they’re here to make programming more accessible and collaborative. So, don’t be afraid to dive in and start experimenting!

**Q&A and Hands-On Activity**

Let’s open the floor for questions. After that, we’ll do a quick hands-on activity where you’ll use an LLM to write a simple program. I’ll guide you through the process step by step.

Вот несколько простых практических заданий по программированию с использованием LLM (больших языковых моделей):

**1. Генерация кода по описанию задачи**

**Задание:** Напишите текстовое описание задачи, например:  
*"Напишите функцию на Python, которая принимает список чисел и возвращает их сумму."*  
Затем попросите LLM сгенерировать код для этой задачи. Запустите код и проверьте его работоспособность.

**2. Оптимизация кода**

**Задание:** Возьмите следующий код и попросите LLM его оптимизировать:

python

CopyEdit

def factorial(n):

if n == 0:

return 1

else:

return n \* factorial(n - 1)

Сравните предложенный LLM вариант с исходным кодом.

**3. Поиск ошибок в коде**

**Задание:** Возьмите код с ошибками и попросите LLM его исправить:

python

CopyEdit

def greet(name):

print("Hello, " + name)

Скопируйте исправленный вариант и запустите его.

**4. Преобразование кода в другой язык**

**Задание:** Напишите код на Python и попросите LLM перевести его на JavaScript. Например:

python

CopyEdit

def add(a, b):

return a + b

Запустите сгенерированный код в онлайн-компиляторе и проверьте его работу.

**5. Генерация тестов для функции**

**Задание:** Напишите простую функцию, а затем попросите LLM сгенерировать для неё тесты. Например, для функции:

python

CopyEdit

def is\_even(n):

return n % 2 == 0

Попросите LLM написать юнит-тесты на Python с использованием unittest или pytest.

**6. Создание SQL-запросов на основе текстового описания**

**Задание:** Дайте текстовое описание запроса, например:  
*"Вывести все записи из таблицы users, где возраст больше 18 лет."*  
Попросите LLM сгенерировать SQL-запрос и выполните его на тестовой базе данных.

**7. Автоматизация повторяющихся задач**

**Задание:** Сформулируйте рутинную задачу, например:  
*"Скачать HTML-страницу и сохранить её в файл."*  
Попросите LLM сгенерировать код для её автоматизации на Python с использованием requests.

Хотите, чтобы я доработал какие-то из этих заданий или добавил новые? 😊

4o

**You said:**

continue with this list

**ChatGPT said:**

ChatGPT

**8. Генерация регулярных выражений**

**Задание:** Опишите шаблон, который нужно найти в тексте, например:  
*"Найти все email-адреса в строке."*  
Попросите LLM сгенерировать соответствующее регулярное выражение (RegEx) и протестируйте его в Python:

python

CopyEdit

import re

text = "Контакты: example@mail.com, test123@gmail.com"

pattern = r"..." # вставьте регулярное выражение, предложенное LLM

matches = re.findall(pattern, text)

print(matches)

**9. Генерация документации к коду**

**Задание:** Напишите код функции без документации, например:

python

CopyEdit

def calculate\_area(radius):

return 3.1415 \* radius \*\* 2

Попросите LLM сгенерировать docstring в формате Google или NumPy и добавьте его в код.

**10. Объяснение кода на естественном языке**

**Задание:** Скопируйте небольшой участок кода и попросите LLM объяснить его построчно. Например:

python

CopyEdit

for i in range(5):

print(i \* i)

Сравните объяснение LLM со своим пониманием.

**11. Генерация псевдокода из кода**

**Задание:** Возьмите любой фрагмент кода и попросите LLM переписать его в виде псевдокода. Например:

python

CopyEdit

def reverse\_string(s):

return s[::-1]

LLM должен преобразовать его в описание алгоритма, например:  
*"Функция принимает строку, переворачивает её, используя срез, и возвращает результат."*

**12. Генерация альтернативных решений**

**Задание:** Дайте LLM код и попросите предложить альтернативный способ его написания. Например:

python

CopyEdit

numbers = [1, 2, 3, 4, 5]

squares = []

for num in numbers:

squares.append(num \*\* 2)

LLM может предложить вариант с list comprehension:

python

CopyEdit

squares = [num \*\* 2 for num in numbers]

Сравните эффективность предложенных решений.

**13. Создание веб-страницы с LLM**

**Задание:** Попросите LLM сгенерировать HTML-страницу с заголовком, параграфом и кнопкой. Например:  
*"Создай простую веб-страницу с заголовком 'Привет, мир!' и кнопкой 'Нажми меня'."*  
Скопируйте сгенерированный код в файл .html и откройте в браузере.

**14. Генерация простого чат-бота**

**Задание:** Попросите LLM сгенерировать Python-код простого текстового чат-бота, например:  
*"Напиши чат-бота, который отвечает на 'Привет' фразой 'Здравствуйте!'"*  
Запустите предложенный код и протестируйте бота.

**15. Генерация данных для тестирования**

**Задание:** Попросите LLM сгенерировать тестовые данные в формате JSON или CSV, например:  
*"Создай 5 записей о пользователях с именем, возрастом и email."*  
Используйте их в коде для тестирования.

**16. Создание GUI-приложения**

**Задание:** Попросите LLM сгенерировать код на Python с использованием tkinter для простого GUI-приложения. Например:  
*"Создай окно с кнопкой, при нажатии на которую выводится сообщение 'Привет!'"*  
Запустите код и убедитесь, что приложение работает.

**17. Создание простого API с Flask**

**Задание:** Попросите LLM сгенерировать код API на Flask, например:  
*"Создай API с одним GET-эндпоинтом /hello, который возвращает JSON {"message": "Привет, мир!"}."*  
Запустите сервер и проверьте ответ API с помощью Postman или curl.

**18. Конвертация JSON в CSV**

**Задание:** Дайте LLM JSON-данные, например:

json

CopyEdit

[

{"name": "Alice", "age": 25},

{"name": "Bob", "age": 30}

]

Попросите его сгенерировать код для их конвертации в CSV. Запустите код и убедитесь, что он работает.

**19. Генерация кода для работы с API**

**Задание:** Попросите LLM написать код для получения данных с публичного API, например OpenWeather. Например:  
*"Напиши код на Python, который получает текущую погоду в Лондоне с OpenWeather API."*  
Запустите код (не забудьте зарегистрировать API-ключ).

**20. Создание телеграм-бота**

**Задание:** Попросите LLM сгенерировать код для простого Telegram-бота с библиотекой python-telegram-bot. Например:  
*"Создай телеграм-бота, который отвечает 'Привет!' на любое сообщение."*  
Настройте токен и запустите бота.