AI-ASSISTED PROGRAMMING

LECTURE 2: INTRODUCTION TO THE FUTURE OF DEVELOPMENT

AI Assisted Programming Course
Duration: 60 minutes

LEARNING OBJECTIVES

- Understand the concept of AI-assisted programming

- Explore current AI tools for developers

- Analyze the impact on productivity and code quality

- Examine real-world adoption statistics

- Discuss benefits and challenges

- Look ahead to the future of programming

WHAT IS AI-ASSISTED PROGRAMMING?

AI-Assisted Programming is the use of artificial intelligence tools to:

‣ Generate code automatically

‣ Complete code as you type

‣ Suggest improvements and optimizations

‣ Debug and fix errors

‣ Translate between programming languages

‣ Generate documentation and tests

AI PROGRAMMING WORKFLOW
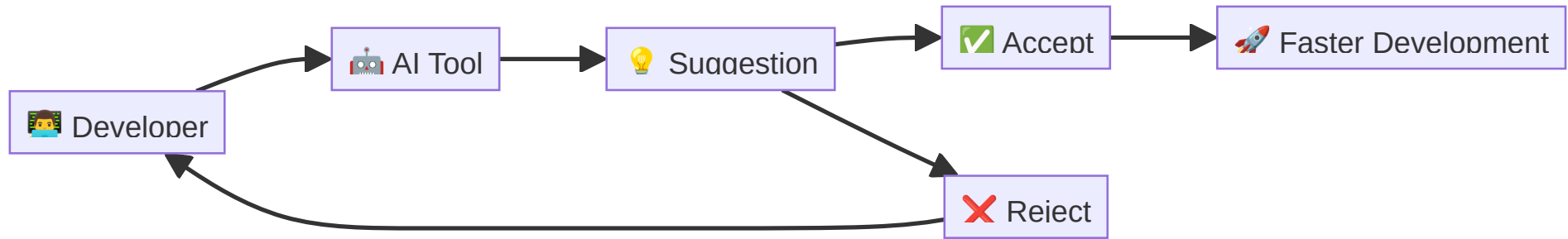
MARKET ADOPTION (2024)

92%
of developers use AI tools

46%
productivity improvement

70%
faster code completion

25%
reduction in bugs

Sources: Stack Overflow Developer Survey 2024, GitHub Research

POPULAR AI PROGRAMMING TOOLS

Tool	Company	Primary Feature	Languages
GitHub Copilot	Microsoft/GitHub	Code completion	40+ languages
ChatGPT/GPT-4	OpenAI	Code generation	All major languages
Claude	Anthropic	Code analysis	All major languages
Tabnine	Tabnine	AI completion	30+ languages

AI DEVELOPMENT ECOSYSTEM

AI Tools Support Every Development Stage

Code Completion

Documentation

Developer

AI Assistants

Code Review

"
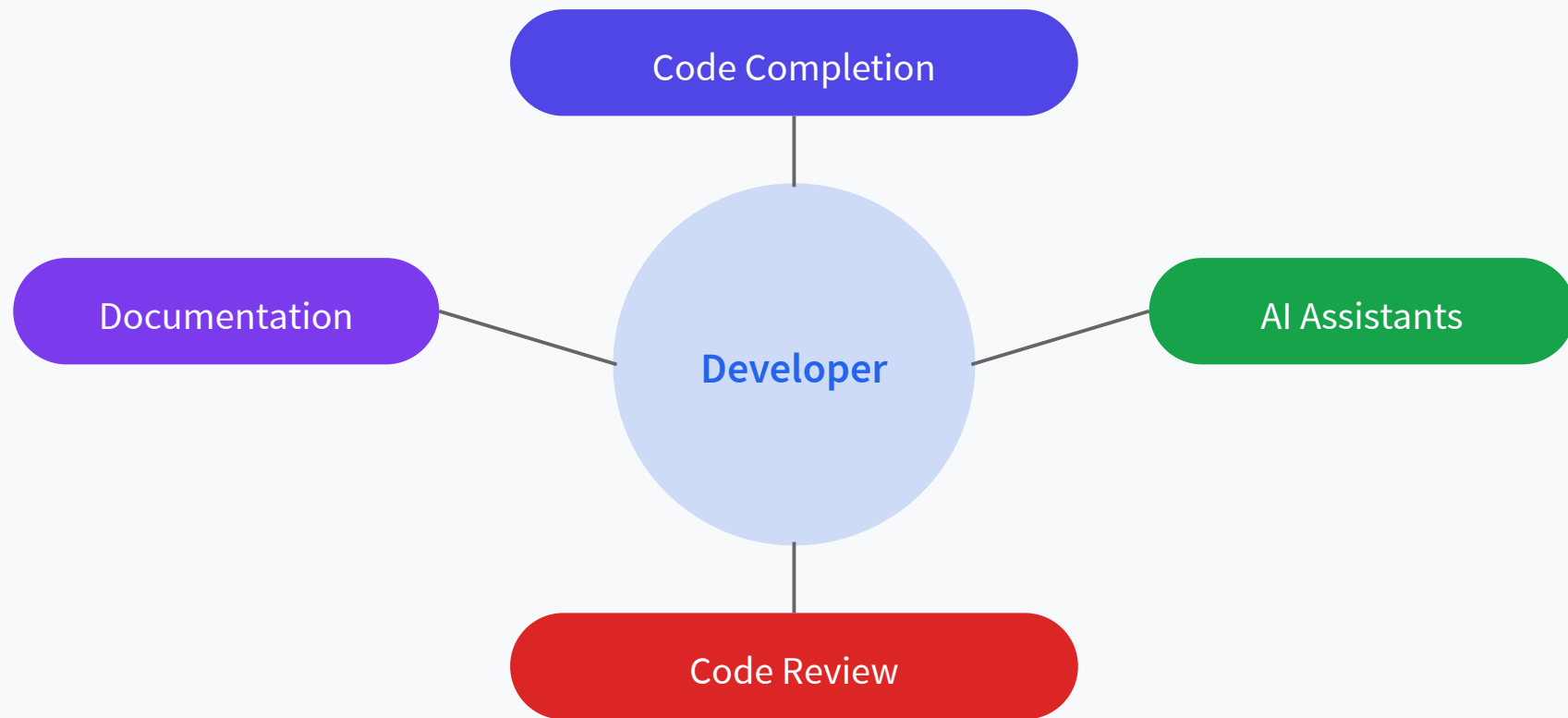
GITHUB COPILOT

The most widely adopted AI programming assistant

‣ Trained on billions of lines of public code

‣ Integrated directly into IDEs (VS Code, JetBrains, etc.)

‣ Real-time code suggestions

‣ Context-aware completions

‣ Chat interface for code explanation

COPILOT CAPABILITIES

✅ STRENGTHS

- Fast code completion
- Understands context
- Learns from comments
- Multiple suggestions
- Wide language support

⚠️ LIMITATIONS

- May suggest incorrect code
- Requires code review
- Limited business logic
- Potential licensing issues

- Internet dependency

OUR LAB ENVIRONMENT: GITHUB CODESPACES

A cloud-based development environment fully configured for our course.

▸ **Instant Setup:** Click "Open in Codespace" and you're ready to go.

▸ **Pre-installed Tools:** Comes with Python, Jupyter, and all necessary extensions.

▸ **Integrated Copilot:** GitHub Copilot is built-in and ready to assist.

▸ **Consistent Environment:** Everyone has the exact same setup, eliminating "it works on my machine" issues.

> *Codespaces provides a managed, on-demand development environment, allowing you to focus on learning, not on setup.*

COMPLETING LABS WITH COPILOT

Follow these steps to complete your first lab:
1. Open the lab by creating a new **Codespace**.
2. Navigate to the lab file (e.g., `setup_lab.py`).
3. Read the `TODO` comments to understand the task.
4. Use **Copilot's suggestions** to help you write the code.
5. **Test your code** using the provided test block.
6. Commit and push your changes to GitHub.

LAB 1: STRUCTURE OVERVIEW

Your first lab will guide you through several common programming tasks with AI assistance.
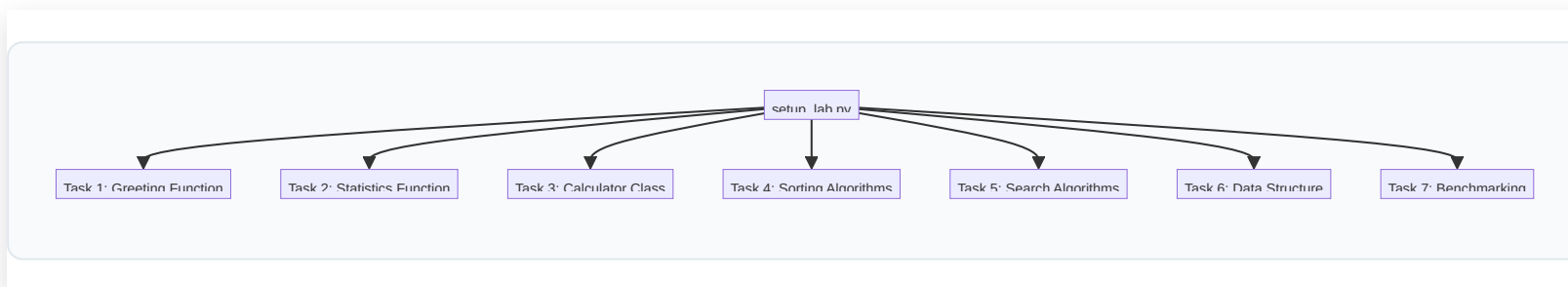
Use the Mermaid diagram to visualize the tasks in the lab file.

LATEST GITHUB COPILOT UPDATES (2025)

Exciting new features and improvements released in September 2025

🚀 MAJOR MODEL UPDATES

- **GPT-5 & GPT-5 mini** - Generally available with enhanced code generation

- **Claude Opus 4.1** - In public preview with improved reasoning

- **Gemini 2.5 Pro** - Available for advanced code analysis

- **Grok Code Fast 1** - Rolling out for faster completions

NEW FEATURES & CAPABILITIES

🤖 AI MODEL SELECTION

- Auto model selection in VS Code
- GPT-4.1 for code completion
- Context-aware model switching
- Performance optimization

🔧 DEVELOPER TOOLS

- Generated commit messages
- Read-only Sparks sharing
- Controlled data access
- Enhanced chat interface

INTEGRATION & ECOSYSTEM

GitHub MCP Registry enables seamless integration with external tools and services

‣ Raycast integration for productivity

‣ VS Code v1.104 with Copilot improvements

‣ Enhanced plugin ecosystem

‣ Better team collaboration features

IMPACT ON DEVELOPMENT WORKFLOW

EXPECTED IMPROVEMENTS (2025)

▸ **Code Quality:** 30% improvement

▸ **Development Speed:** 40% faster

▸ **Debugging Time:** 35% reduction

▸ **Learning Curve:** 50% reduction

▸ **Code Review:** 25% faster

▸ **Documentation:** 45% improvement

Projected based on new model capabilities and features

LIVE DEMO: AI CODE GENERATION

```javascript
// Comment: Create a function to calculate fibonacci numbers
function fibonacci(n) {
    if (n <= 1) return n;
    return fibonacci(n - 1) + fibonacci(n - 2);
}


// Comment: Create an optimized version with memoization
function fibonacciMemo(n, memo = {}) {
    if (n in memo) return memo[n];
    if (n <= 1) return n;
    memo[n] = fibonacciMemo(n - 1, memo) + fibonacciMemo(n - 2, memo);
    return memo[n];
}


// Comment: Generate test cases
console.log(fibonacci(10)); // Expected: 55
console.log(fibonacciMemo(50)); // Much faster for large numbers
```

Example of AI-generated code with improvements

PRODUCTIVITY IMPACT

DEVELOPER TASK TIME REDUCTION

▸ Code writing: **55% faster**

▸ Bug fixing: **37% faster**

▸ Code review: **30% faster**

▸ Documentation: **60% faster**

▸ Testing: **45% faster**

▸ Refactoring: **40% faster**

▸ Learning new APIs: **65% faster**

▸ Debugging: **35% faster**

Source: GitHub Copilot Research Study 2024

KEY BENEFITS

🚀 FOR DEVELOPERS

- Faster coding and reduced boilerplate
- Learning new languages and frameworks
- Reduced context switching
- Enhanced creativity and problem-solving

🏢 FOR ORGANIZATIONS

- Increased development velocity
- Reduced time-to-market
- Lower training costs
- Improved code consistency

CHALLENGES & CONSIDERATIONS

⚠️ TECHNICAL CHALLENGES

- Code quality and correctness
- Security vulnerabilities
- Over-reliance on AI suggestions
- Debugging AI-generated code

🔒 ETHICAL & LEGAL

- Code ownership and licensing
- Privacy and data security
- Bias in AI models
- Impact on developer skills

BEST PRACTICES

‣ **Always review AI-generated code**

‣ **Write clear comments** to guide AI suggestions

‣ **Test thoroughly** - AI code may have subtle bugs

‣ **Understand the code** before accepting suggestions

BEST PRACTICES (CONTINUED)

- **Use AI as a tool, not a replacement** for thinking
- **Stay updated** on security and licensing implications
- **Maintain coding skills** alongside AI usage
- **Consider team consistency** in AI tool usage

FUTURE: EMERGING TRENDS

▸ More specialized AI models for specific domains

▸ Better integration with development workflows

▸ AI-powered code review and testing

▸ Natural language to code translation

▸ Automated refactoring and optimization

FUTURE: IMPACT ON DEVELOPERS

‣ Focus shifts to higher-level problem solving

‣ Increased importance of code review skills

‣ Need for AI literacy in development

‣ Emphasis on creative and architectural thinking

‣ Continuous learning becomes more critical

THIS COURSE PREVIEW

UPCOMING LECTURES:

- Code Generation & Completion

- Code Review & QA

- Testing & Debugging

- Documentation

WHAT YOU'LL LEARN:

- Hands-on tool usage

- Best practices

- Real-world applications

- Ethical considerations

QUESTIONS & DISCUSSION

What questions do you have about AI-assisted programming?
DISCUSSION TOPICS:

‣ Have you used AI programming tools before?

‣ What concerns do you have about AI in development?

‣ Which tools are you most excited to learn about?

THANK YOU!

INTRODUCTION TO AI-ASSISTED PROGRAMMING

Speaker notes