Intro to Python & AI

Parent Preview & Program Overview

Haytham Institute Summer Program

Program Duration: 3 weeks (virtual format)
Target Audience: Ages 10-15 (middle school students)
Meeting Schedule: 5 sessions per week, 120 minutes each (2 hours)
Prerequisite: Pre-algebra and algebra (variables, equations, graphing)
No programming experience required!

Program Overview

Intro to Python & AI (IPAI) is an exciting hands-on introduction to programming and artificial intelligence! Students will learn Python from scratch through engaging projects like games, quizzes, and creative applications. They'll then discover how machines learn, train their own AI models, and explore real-world AI applications. By the end, students will build original machine learning projects and understand how AI is shaping our world.

This program emphasizes learning by doing—students code from day one and build progressively sophisticated projects. No boring lectures—just hands-on coding, creative problem-solving, and fascinating AI experiments!

What Students Will Learn

By the end of IPAI, students will be able to:

- Write Python programs using variables, conditionals, loops, functions, and data structures
- Create interactive applications including games, quizzes, calculators, and chatbots
- Understand machine learning fundamentals: training, testing, and prediction
- Train AI models to recognize images, sounds, and patterns using professional ML tools
- Explain different types of machine learning and their real-world applications
- Discuss AI ethics, bias, and responsible technology development
- Build and present an original AI-powered project
- Debug code, solve programming problems, and think computationally

Weekly Themes

Week 1: Python Fundamentals Through Games and Apps

Central Questions: What is programming? How do computers follow instructions? How can we build interactive programs?

Key Topics:

- Python basics: variables, input/output, data types
- Conditional statements and decision-making logic
- Loops for repetition and iteration
- Functions and code organization
- Lists and data collections

Project Highlights:

- Mad Libs story generator with user input
- Password checkers and Magic 8 Ball fortune teller
- Number guessing game with hints and score tracking
- Rock Paper Scissors game with multiple functions
- Interactive quiz application with score management
- To-do list and contact management systems

Skills Developed:
Students learn fundamental programming concepts through immediate application. Every session includes hands-on coding with projects students can share and use. Emphasis on writing clean code, debugging errors, and thinking algorithmically.

Take-Home Activities:
Students complete creative coding challenges that reinforce daily concepts, often involving family members as testers for their programs.

Week 2: Advanced Python & Introduction to AI

Central Questions: How do we organize complex data? What makes machines "intelligent"? How do computers learn from examples?

Key Topics:

- Dictionaries and advanced data structures
- File handling and data persistence
- Introduction to Artificial Intelligence and Machine Learning
- Training and testing ML models

- Types of machine learning: supervised, unsupervised, reinforcement

Project Highlights:

- Student databases and character management systems
- Simple chatbots with pattern matching
- Persistent applications with file storage (to-do lists, journals, high score trackers)
- Image classifiers using Google's Teachable Machine
- Hand gesture and sound recognition models
- Experiments measuring the relationship between training data and accuracy

Skills Developed:
Students advance to professional data structures and learn to save/load information from files. Second half introduces AI fundamentals through hands-on training of machine learning models. Students experience how computers learn from examples rather than explicit programming.

Take-Home Activities:
Students create custom ML models, research AI in daily life, and design experiments exploring machine learning principles.

Week 3: Building AI Projects and Exploring the Future

Central Questions: How do neural networks work? How do we build responsible AI? What can we create by combining Python and ML?

Key Topics:

- Neural networks and deep learning fundamentals
- Collecting quality training data and model iteration
- AI ethics, bias, privacy, and accountability
- Integrating ML models with Python applications
- Building complete AI-powered projects

Project Highlights:

- Human neural network simulation and interactive neuron exploration
- Custom-trained models with systematic data collection and testing
- Bias demonstrations showing importance of balanced datasets
- Gesture-controlled games combining ML with Python code
- AI-powered chatbots with intent recognition
- Image classifier applications for practical problems
- Final capstone projects integrating all learned skills

Skills Developed:
Students understand how neural networks process information and make predictions. They learn the complete ML pipeline: data collection, training, evaluation, and deployment. Critical discussions about AI ethics prepare students to develop technology responsibly. Final projects synthesize Python programming with machine learning.

Take-Home Activities:
Students train custom models, research AI ethics cases, interview family about AI, and complete comprehensive final projects demonstrating both coding and ML skills.

Teaching Approach

CodeCamp uses project-based learning where students code from day one. Our approach includes:

Hands-On Coding:

- Interactive programming exercises with immediate feedback
- Progressive complexity building on previous concepts
- Real projects students can share and use
- Professional tools (Python, Teachable Machine, IBM frameworks)

Age-Appropriate Engagement:

- Concepts explained through relatable analogies and examples
- Game-based learning and competitions
- Visual demonstrations and interactive simulations
- Emphasis on creativity and problem-solving over syntax memorization

AI Ethics Integration:

- Critical discussions about bias, fairness, and responsibility
- Hands-on demonstrations of AI limitations and failures
- Real-world case studies
- Development of ethical guidelines for technology use

Progressive Complexity:

- Week 1: Master Python fundamentals through games and applications
- Week 2: Advance to complex data structures and introduction to machine learning
- Week 3: Understand neural networks, train custom models, build AI-powered applications

Primary Learning Platforms

Students will use professionally-developed tools and environments:

Programming Environments:

- Replit or Thonny (beginner-friendly Python editors)
- VS Code (optional, for more advanced students)
- Google Colab (for collaborative coding)

AI/ML Tools:

- Google Teachable Machine - Train image, sound, and pose models
- TensorFlow Playground - Visualize neural networks learning
- Carnegie Mellon Neuron Sandbox - Interactive neuron exploration
- Machine Learning for Kids - Additional practice projects

All platforms are free and browser-based when possible to minimize technical barriers.

Assessment & Completion

This is an enrichment program focused on skill development and creativity rather than traditional grading. Success is measured through:

- **Active participation** during coding sessions (25%)
- **Daily project completion** demonstrating concept mastery (40%)
- **Take-home challenges** reinforcing learning (20%)
- **Final project** showcasing integrated Python and AI skills (15%)

Completion Requirements:

- Attend at least 13 of 15 sessions
- Complete at least 12 of 15 daily programming projects
- Submit at least 10 of 15 take-home challenges
- Complete and present final project

Note: Emphasis is on effort, growth, and creative problem-solving. Bugs and errors are valuable learning opportunities, not failures!

Final Project Options: Students choose from smart home controllers, educational games with ML, environmental helper apps, accessibility tools, creative AI generators, or propose their own ideas combining Python and machine learning.

Required Materials

Hardware:

- Computer with internet connection (Windows, Mac, or Chromebook)
- Webcam (for training image-based ML models)
- Microphone (for sound-based ML projects)
- Headphones recommended for clear communication

Software (Set up during first session):

- Python 3.8+ and code editor (Replit, Thonny, Google Colab, etc.)
- Modern web browser (Chrome recommended)
- Video conferencing software (Zoom or equivalent)
- Free accounts: Replit, Teachable Machine

Optional:

- Notebook for planning projects and sketching algorithms
- Second monitor/screen for following along while coding

All platforms and tools are completely free. No paid subscriptions required.

Beyond our camp

Students will leave with resources to continue their coding and AI journey:

Continuing Exploration:

- Free Python learning platforms (Python.org, Real Python, FreeCodeCamp)
- AI/ML resources (Machine Learning for Kids, Elements of AI course)
- YouTube channels for young programmers
- Recommended books: "Python for Kids," "Teach Your Kids to Code"
- Project ideas for independent development

Future Pathways:

- Foundation for advanced programming courses and competitions
- Understanding of AI/ML concepts relevant to many career fields
- Computational thinking skills applicable across disciplines
- Confidence building technology solutions to real problems
- Awareness of emerging career opportunities in AI and software development

Next-Level Learning:

- Advanced Python courses (algorithms, data structures, web development)

- Deeper ML courses (Coursera, Kaggle, Fast.ai)
- Project-based platforms for ongoing practice
- Coding communities and competitions

Course Policies

Attendance

Regular attendance is essential as each session builds on previous concepts. If you must miss a session, notify the instructor in advance, review recorded materials (if available), and complete coding exercises independently.

Collaboration and Academic Integrity

Encouraged:

- Helping classmates debug code and solve problems
- Sharing approaches and discussing solutions
- Working together during class activities
- Explaining concepts to each other

Not Allowed:

- Copying code from classmates for individual assignments
- Submitting others' work as your own
- Using AI code generators without understanding the code

Best Practice: Research syntax and examples online, but type code yourself and ensure you understand every line.

Communication

- **During sessions:** Use raised hand or chat features
- **Outside sessions:** Email instructor or use course platform
- **Technical issues:** Contact instructor immediately
- **Response time:** Within 24 hours on weekdays

Accessibility

We are committed to making coding accessible to all students. Contact the instructor during Week 1 if you need accommodations for learning differences, have technical limitations, or require additional support.

Program Philosophy

Learning to code should be project-based, creative, and empowering.

IPAI is built on these principles:

Project-Based Learning: Every concept is immediately applied to real projects students can use and share. No abstract exercises—build games, apps, and AI models from day one.

Error-Friendly Environment: Bugs are valuable learning opportunities, not failures. We celebrate debugging as an essential programming skill.

Creative Expression: Programming is creative problem-solving. There are many correct solutions. We value unique approaches and innovative thinking.

Real-World Connections: Focus on applications students care about—games, chatbots, practical AI tools. Abstract concepts tied to concrete, meaningful uses.

Collaborative Learning: Programmers work in teams. Students learn from each other, help debug together, and celebrate shared successes.

Ethical Technology: With coding power comes responsibility. We discuss AI ethics, bias, privacy, and commit to developing technology for positive impact.

Ages 10-15 is the perfect time to learn programming because students have the logical thinking skills for coding, the creativity for innovative solutions, and the digital fluency to see technology's real-world relevance.

Parent Information

What to Expect Each Week

Week 1: Your child will write Python code from day one—games, quizzes, and interactive apps. They'll likely want to test their programs with you!

Week 2: Advanced programming combined with AI introduction. Your child will train machine learning models using their webcam to recognize gestures, faces, and sounds.

Week 3: Final project week. Your child will combine Python skills with machine learning to build an original AI-powered application.

How You Can Support

- Provide quiet space and reliable internet during sessions
- Test their programs and provide enthusiastic feedback
- Let them explain concepts (teaching reinforces learning!)
- Encourage persistence when code doesn't work (debugging is essential!)
- Celebrate projects regardless of complexity

Skills Your Child Will Gain

Beyond coding, students develop:

Problem-Solving: Breaking complex challenges into manageable steps
Logical Thinking: Understanding cause-effect relationships and conditional reasoning
Persistence: Debugging teaches resilience and systematic troubleshooting
Creativity: Programming as creative expression and innovative problem-solving
Critical Thinking: Evaluating AI systems, recognizing bias, considering ethical implications
Digital Literacy: Understanding how technology they use daily actually works

After IPAI

Programming and AI literacy are increasingly valuable across all career fields—from medicine to art, business to science. This camp provides the foundation and resources for continued independent learning, including free platforms, project ideas, and recommendations for more advanced courses.