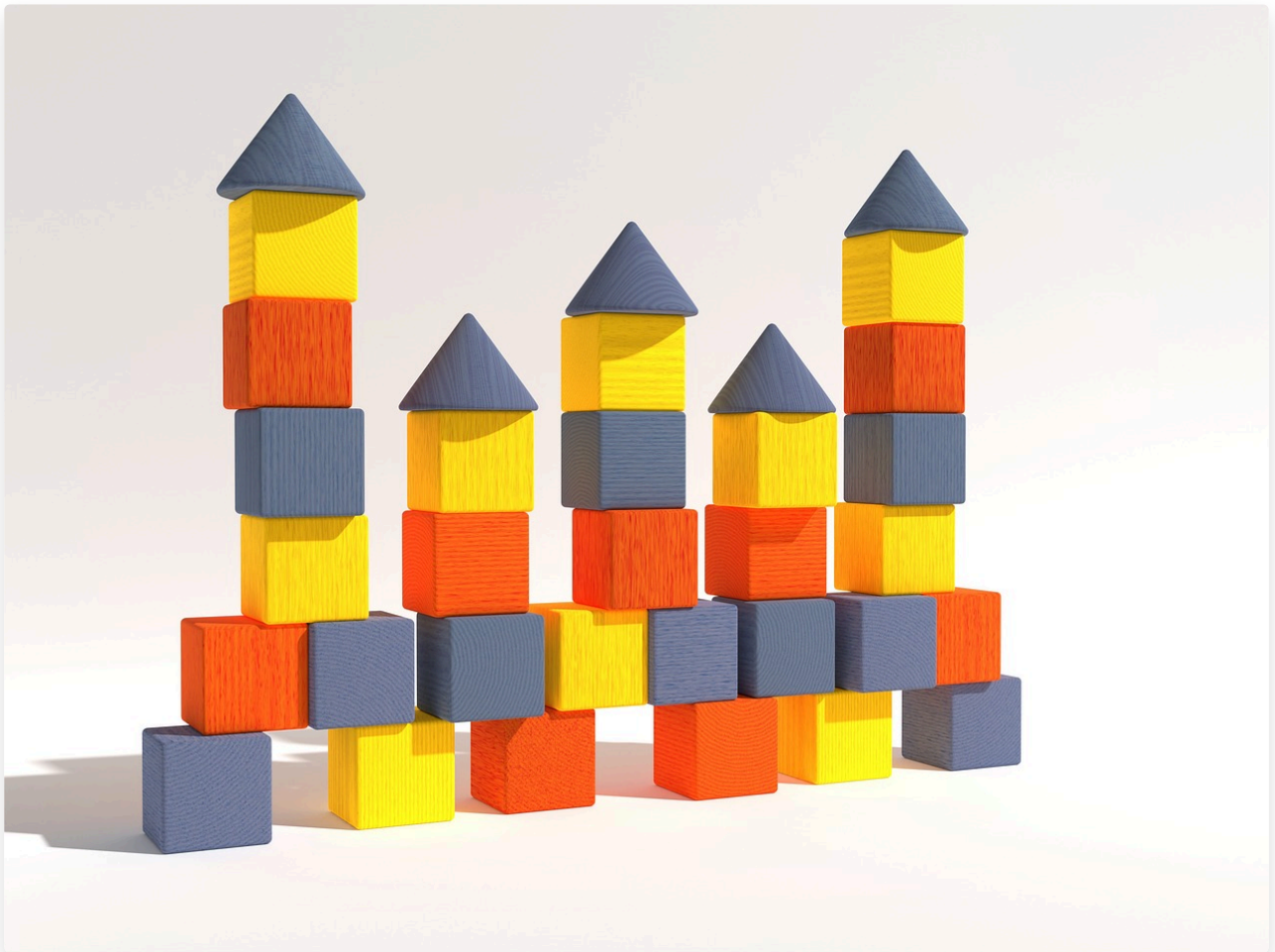


From Patterns to Programs

Textbook

From Patterns to Programs



Introduction

This explores how beginner-friendly block programming connects to the deeper concepts of loops, conditions, functions, and even object-oriented programming. You'll discover how real-world problems can be solved by thinking like a programmer—step by step, block by block!

What is Block Programming?

Block programming is a way to write code using colorful puzzle-like pieces instead of typed words. These blocks snap together to form commands that a computer can follow. This method helps beginners focus on logic without worrying about spelling or syntax. You can use blocks to create loops (repeating actions), conditionals (if-then choices), and even write functions (custom actions) that make your code easier to understand and reuse!

Loops and Logic: Iterative and Non-Iterative Structures

Some actions in programming happen **over and over again**. These repeated actions are called **iterative structures**, and they often use **loops**—like **for** loops and **while** loops. Other times, you only want an action to happen **once**, like a single calculation or a message that displays at the start. That's a **non-iterative** structure.

A famous and fun example of using loops in programming is the **Fibonacci sequence**. This sequence starts with 0 and 1, and every number after that is the **sum of the two numbers before it**:

0,1,1,2,3,5,8,13,21

This pattern naturally fits an **iterative structure**, because the process of adding two numbers to make the next one **repeats**. You can write a loop that continues generating Fibonacci numbers as long as you want—just tell the computer how many terms to calculate or set a maximum value.

Why does this matter? Because writing a program to generate the Fibonacci sequence helps you understand:

- **How loops store and reuse data**
- **The power of iteration to solve complex patterns**
- **How mathematical logic is translated into code**

In fact, some programmers use **recursive functions** (a type of non-iterative structure that calls itself) to calculate Fibonacci numbers too. Comparing **iterative** and **recursive** solutions for the same problem can show which is faster or more efficient—a great lesson in coding strategy!

Learning when to **repeat**, when to **stop**, and how to **track values across steps** is a critical part of writing **efficient and elegant code**—and the Fibonacci sequence is a perfect way to practice.

Expressions, Variables, and Conditional Statements

Expressions are like math problems in your code — they combine numbers and values to get a result. A **variable** is a container that stores information, like a score or a name. And **conditional statements** let your program make decisions. For example: *If the player has 0 lives, then end the game.* These tools help your program respond to different situations in smart and flexible ways.

Introducing Classes in Object-Oriented Programming

As you grow in coding skills, you'll start to explore object-oriented programming. A **class** is like a blueprint. It tells the computer how to build something called an **object**. For example, you could make a class called "Dog" with properties like color, size, and breed. Every dog object you create from that class will follow the same blueprint, but with its own unique details.

Why Classes Matter in Your Code

So why do we use classes? Because they help us stay organized. Instead of rewriting the same instructions over and over, we write the class once and use it as many times as we want. That means fewer mistakes, cleaner code, and faster programs. Whether you're making a game or designing an app, using classes helps your program grow as your ideas get bigger!

Critical Thinking Questions

1. Why might a programmer choose to use a loop instead of copying and pasting the same block of code several times?
2. Imagine you are designing a game. What kind of class might you create, and what properties or actions would your class include?
3. If you were explaining variables and conditionals to someone new to programming, what real-life

comparison could you use to help them understand these ideas?

Questions (5)

1. What is one of the main benefits of using block programming for beginners?

MULTIPLE CHOICE

Choose the correct answer:

- A. It only works for video games
- B. It helps focus on logic without worrying about typing or syntax
- C. It uses only math-based commands
- D. It requires knowledge of object-oriented programming

2. What is an iterative structure in programming?

MULTIPLE CHOICE

Choose the correct answer:

- A. A command that only runs one time
- B. A decision made by a conditional statement
- C. A process that repeats, like a loop
- D. A part of a class

3. Which of the following is a good example of a variable in a video game?

MULTIPLE CHOICE

Choose the correct answer:

- A. The player's background music
- B. The size of the screen
- C. The player's current score
- D. A sound effect when a button is clicked

4. What is the purpose of a class in object-oriented programming?

MULTIPLE CHOICE

Choose the correct answer:

- A. To organize blueprints for objects
- B. To create one-time instructions
- C. To avoid using any variables
- D. To make graphics for a game

5. Which of these actions is best handled with a conditional statement?**Choose the correct answer:**

- A. Displaying your game character
- B. Adding a new level to your game
- C. Moving the character every second
- D. Playing a sound when the player wins

Answer Keys & Solutions

Questions

1. What is one of the main benefits of using block programming for beginners?

MULTIPLE CHOICE

Correct Answer:

- A. It only works for video games ✗ Incorrect
- B. It helps focus on logic without worrying about typing or syntax ✓ Correct
- C. It uses only math-based commands ✗ Incorrect
- D. It requires knowledge of object-oriented programming ✗ Incorrect

2. What is an iterative structure in programming?

MULTIPLE CHOICE

Correct Answer:

- A. A command that only runs one time ✗ Incorrect
- B. A decision made by a conditional statement ✗ Incorrect
- C. A process that repeats, like a loop ✓ Correct
- D. A part of a class ✗ Incorrect

3. Which of the following is a good example of a variable in a video game?

MULTIPLE CHOICE

Correct Answer:

- A. The player's background music ✗ Incorrect
- B. The size of the screen ✗ Incorrect
- C. The player's current score ✓ Correct
- D. A sound effect when a button is clicked ✗ Incorrect

4. What is the purpose of a class in object-oriented programming?

MULTIPLE CHOICE

Correct Answer:

- | | |
|---------------------------------------|-------------|
| A. To organize blueprints for objects | ✓ Correct |
| B. To create one-time instructions | ✗ Incorrect |
| C. To avoid using any variables | ✗ Incorrect |
| D. To make graphics for a game | ✗ Incorrect |

5. Which of these actions is best handled with a conditional statement?

MULTIPLE CHOICE

Correct Answer:

- | | |
|---|-------------|
| A. Displaying your game character | ✗ Incorrect |
| B. Adding a new level to your game | ✗ Incorrect |
| C. Moving the character every second | ✗ Incorrect |
| D. Playing a sound when the player wins | ✓ Correct |