

1. Previous Learning & Learning Objectives

Previous Learning (What You Already Know)

- **Programming Languages:** Programming languages come in block-based forms (like Scratch) and text-based forms (like Python), where users enter commands into an Integrated Development Environment (IDE) to translate and run code.
- **Real-World Libraries:** A library stores resources for people to access, borrow, and use (such as a school library with books).
- **Sub-routines:** A self-contained algorithm with an identifier (name). When called by its identifier, the program branches to execute the statements inside the sub-routine and returns to the calling point once finished.

Learning Objectives

- Understand why program libraries exist and what they contain.
- Know how to import a library into a Python program using the import command.
- Call library functions and pass required parameters.
- Understand how to capture, store, or output return values from functions.
- Work with key functions from the math library (floor, ceil, pi) and the random library (randint, uniform).

2. Classroom Discussion and Core Concepts

A. Sub-routines: Functions vs. Procedures

- **Sub-routine:** A named, self-contained algorithm.
- **Function:** A sub-routine that calculates and returns a value back to the calling program using a keyword like return.
- **Procedure:** A sub-routine that executes a sequence of instructions but does not return a value.

Translating Checkpoint (In My Own Words):

- *Function:* A mini-program that does a job and hands back a result (like asking a friend for the answer to a math problem).
- *Procedure:* A mini-program that performs an action but keeps its hands empty when done (no returned answer).

Python Function Syntax & Rules

def functionName():

lines of code that form the function body
lines of code that form the function body
lines of code that form the function body
lines of code that form the function body
return result

- Functions begin with the keyword def, followed by a valid identifier, empty or parameterized parentheses (), and a colon :.

- **Identifier Rules:** Must start with a letter and contain only one word without spaces (e.g., myFunction1 is valid; 123 and my function are invalid).
- **Indentation:** All lines belonging to the function body must be indented.

B. Parameters and Return Values

- **Parameter:** A value sent into a sub-routine inside parentheses when the sub-routine is called, providing data the sub-routine needs to execute its task (e.g., squareArea(length)).
- **Return Value:** The value sent back to the main program by the function. In execution, the function call is replaced by its return value, which can be printed immediately or assigned to a variable.

Contextualizing Scenario: Why do we need parameters? A function designed to calculate the area of a square cannot work unless it knows how long the sides are. Passing the length as a parameter gives the function the exact measurement it needs each time it is used.

C. Programming Language Libraries or Builtin Libraries

- **Definition:** A collection of pre-written sub-routines created by programmers and made available for others to reuse.
- **Why Use Them?**
 1. *Saves Time:* Reusing the code rather than rewriting common formulas from scratch.
 2. *Avoids Errors & Reliable:* Library programs undergo extensive testing with large ranges of data.
 3. *Focus on core logic:* Allows developers to focus on the core logic.
- **Drawbacks & Risks:**
 - Anyone can publish a library, meaning untrusted libraries may carry code that could damage a computer (such as viruses).
 - Even tested libraries need to be verified to ensure they return the required result.
- **Import Syntax:** Place import <library_name> at the top of the program.

3. Q&A (Short Answer Questions)

Q1: What is a program library?

A program library is a collection of pre-written sub-routines that programmers make available for others to use in their programs. It acts like a code repository that saves developers from having to write common routines themselves.

Q2: State two main reasons why programmers use program libraries.

Programmers use libraries to save valuable time and to prevent programming errors. Library functions have undergone extensive testing across broad datasets, making them reliable and efficient.

Q3: What kind of risk is associated with downloading external program libraries?

Because anyone can publish a program library, downloading from untrusted sources carries the risk of installing malicious code, such as computer viruses, that can damage a computer system.

Q4: Explain the difference between a function and a procedure.

A function is a sub-routine that sends a calculated value back to the main program using a command like return. A procedure runs a sequence of instructions to carry out a task but does not return any value back to the program.

Q5: What is a parameter in programming?

A parameter is a value sent to a sub-routine inside parentheses when the sub-routine is called. It provides the sub-routine with the specific data values required to execute its instructions.

Q6: What does the Python keyword `def` signify?

The keyword `def` signals to Python that you are defining a new sub-routine or function. It must be followed by a valid function identifier, parentheses, and a colon.

Q7: State two naming rules for a Python function identifier.

A function identifier must start with a letter and must consist of only a single word with no spaces. It cannot begin with a number.

Q8: What does the `math.floor()` function do?

The `math.floor()` function takes a number as a parameter and rounds it down to the nearest whole number. For example, `math.floor(4.55)` returns 4.

Q9: How does `math.ceil()` differ in functionality from `math.floor()`?

While `math.floor()` always rounds a decimal down to the nearest whole number, `math.ceil()` rounds a number up to the next whole number. For instance, `math.ceil(3.1)` evaluates to 4, whereas `math.floor(3.1)` evaluates to 3.

Q10: What is special about the syntax when accessing `math.pi` compared to other functions?

The value of `pi` is called without using brackets after the identifier (`math.pi`), unlike functions such as `math.floor()` or `random.randint()`, which require parentheses.

Q11: How does the `random.randint()` function determine its output?

`random.randint(smallest, largest)` accepts two integer parameters representing lower and upper bounds. It returns a random whole number chosen by chance within and including those boundaries.

4. Second-Order Thinking Skills Questions

Q1 A student asserts: *"Using program libraries in school assignments is cheating because you are taking code that someone else wrote."* Evaluate this statement based on your understanding.

Professional programmers often use ready-made libraries because they save time and help avoid mistakes. In school, students can use library functions when allowed. However, copying code from the internet and saying it is your own work is considered cheating and is not the same as using a library.

Q2 Construct a complete Python program that imports the `random` library and simulates rolling a standard six-sided die, outputting the result.

```
import random
```

```
# Generate a random integer between 1 and 6 inclusive
```

```
die_roll = random.randint(1, 6)  
print("You rolled a:", die_roll)
```

Explanation: `import random` loads the library, and `randint(1, 6)` supplies a random integer between 1 and 6, representing the faces of the die.

5. Higher Order Thinking Skills (HOTS) Questions

Q1 (Analyze & Compare): Look at the following function calls:

- Call A: `random.randint(-2, 1)`
- Call B: `random.randint(100, 103)`

Predict every possible integer return value for each call, and explain why a decimal such as -1.5 or 101.25 cannot be returned.

Answer:

- Call A possible outputs: -2, -1, 0, 1
- Call B possible outputs: 100, 101, 102, 103
- Reason: The `randint()` function exclusively generates and returns whole numbers (integers) between the specified bounds; decimal (real) numbers require a different function such as `uniform()`.