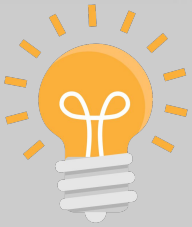

Unit 7: Functions

Unit 6 Reflection

to divider

Unit Summaries

- Unit 1: Beginning in Computer Science
  - Key Concepts: Introduction to Python, basic programming concepts like variables, data types, input/output, and understanding binary.
  - Main Idea: Establish a foundational understanding of computer science principles and Python syntax.
- Unit 2: Number Calculations and Data
  - Key Concepts: Arithmetic operations, data manipulation, working with built-in functions, and understanding the history of computing.
  - Main Idea: Develop skills for performing calculations and data handling in Python.
- Unit 3: Making Decisions
  - Key Concepts: Conditional logic using if-else statements, logical operators, and basic algorithm design.
  - Main Idea: Learn how to implement decision-making processes in programming.
- Unit 4: Repetition and Loops
  - Key Concepts: Using loops for iteration, control variables, and range functions.
  - Main Idea: Master the concepts of loops to automate repetitive tasks in code.
- Unit 5: Programming in EarSketch
  - Key Concepts: Integrating music and programming, using EarSketch to create digital audio compositions.
  - Main Idea: Familiarize yourself with Earsketch code in preparation for sound-related projects
- Unit 6: Graphics
  - Key Concepts: Drawing and manipulating graphics using Python libraries.
  - Main Idea: Explore how to create and control graphics through programming.

Next Page —>

Unit 6 Reflection

to divider

Challenges and Interests

- Most Challenging Topics: Animations, variables in frames, and Earsketch because they're very different from the Python I'm used to and have a lot of depth to them.
- Most Interesting Topics: EarSketch because you can compose music with code and animations because you can make moving art with even a few lines of code.

Grades and Feedback

- Right now, the only thing affecting my grades is notebook checks. I assume it's because I'm thinking of the notebook as a notebook and not a journal so I lose out points in that aspect. Other than that, I'm doing good in other areas.

  Action Steps:

  1. Regard the notebook as more of a journal
  2. Allocate regular study sessions for practicing challenging concepts.
  3. Use online resources or seek help from peers or instructors for difficult topics.
  4. Start early on assignments and projects to manage time effectively.

My goal over these next two weeks is to finish all my school and personal assignments so I can come back to school with no late or pending work. I can do this by allocating time daily to work on assignments so they get done and time isn't wasted. I will also break down assignments into smaller tasks with deadlines to stay on track.

Game Development:

In preparation to this game, I will research how to apply my current skills to the game. For example, how to incorporate Earsketch into the game. My initial game idea is to do something fantasy related where the user has to find objects based on clues to move to the next level. In Unit 7, I would like to learn len, break, and pass so I can use them in assignments.

Unit 7 Notes: 12/19/2024

to divider

Lists:
A list statement is very similar to a variable statement - you assign a list using an equal sign.
A list is a collection of items in a particular order.

- Since a list is usually contains more than one item, it is a good idea to name the list with plural names like: items, digits, or letters
- In Python, square brackets [] indicate a list
- Each item in a list is separated by commas.
- A list can have multiple data types (e.g. int, string, etc.) within a single list.

First number in a range is the start index and second is where you stop

7.1 - 7.4: 1/2/2025

to divider

A function is a collection of commands that are given a name.

- Some built-in functions are print, min, max, and input.
- Functions are used to simplify code, for actions we want to reuse, to organize longer or more difficult programs, and to help with debugging.

In order to create a function, you need to use the keyword **def** which allows you to define a set of code.

- The name of your function should make sense based on the purpose of your function.
- For a function to run, you have to call it. To call it, you reference the function by its name with a pair of parentheses afterwards.

At the top of a program is where you input your modules. Defined functions typically go after the modules. After this, is where the program starts.

An argument is the extra piece of information that goes into a function call that also gets passed to that function.

- For example, with print(), everything between the parentheses is an argument and is printed in the output.
- To receive arguments, a function needs parameters when defined.
  - The parameter is the variable included in the function definition that accepts the value of the argument.
- Common string functions: upper(), lower(), capitalize(), find(), and count() which all return values.
- To return a value means to send a value back to the original function call.
- capitalize() - capitalizes the first letter of the string
- find() - returns the position of the first appearance of a value in the string; returns -1 if the value is not found (the first character in the string is in position 0)
- isdigit() - returns true if a string is all numbers
- lower() - changes the string to all lowercase letters
- replace() - replaces a specific value in a string with a specified value
- upper() - changes the string to all uppercase letters

7.5 - 7.6: 1/2/2025

to divider

Complex programs utilize multiple functions - some of which are called or chained with another function.

- Local variables are used inside a function - once the function ends, those variables and their values are no longer accessible in the program.
- Global variables can be used anywhere in a function or program. You have to use the keyword **global** to specify that.

An important tool in programming is tracing code. It allows you to quickly find errors in your own code and understand what other coders are trying to do in their code.

Unit 7 Conclusion

to divider

Key Takeaways

Function Basics

Functions are collections of commands given a name, used to simplify code, reuse actions, organize complex programs, and aid in debugging. They are defined using the **def** keyword, followed by a descriptive name that reflects the function's purpose.

Parameters and Arguments

Functions can accept extra information through arguments, which are passed to the function as parameters. This allows for more flexible and reusable code.

Returning Values

Many functions return values, sending data back to the original function call. This is crucial for performing operations and obtaining results.

String Manipulation

The unit introduces several built-in string functions like upper(), lower(), capitalize(), find(), and count(), which are useful for text processing.

Variable Scope

Understanding the difference between local and global variables is important. Local variables are confined to the function, while global variables can be accessed throughout the program.

Code Tracing

The unit emphasizes the importance of code tracing as a valuable skill for debugging and understanding other programmers' code.