Computer Science

18 Lessons

Upper Secondary|Grades 9-12|Ages 14-18

Programming with Python

EDUCATION.MINECRAFT.NET

Table of Contents

Getting Started with Minecraft

Install Minecraft: Education Edition

Minecraft: Education Edition can be installed on Chromebook, iPad, Mac, and PC. To ensure your experience with Minecraft: Education Edition is top-notch, make sure your devices meet the minimum system requirements. To check if your device supports Minecraft: Education Edition, see System Requirements.

If you have questions about setting up Minecraft: Education Edition, the following link will provide you with some frequently asked questions and additional information about set-up. On this page, you will find assistance for:

- Get Started
- Purchase Licenses
- Administration and License Management
- Installation
- Troubleshooting

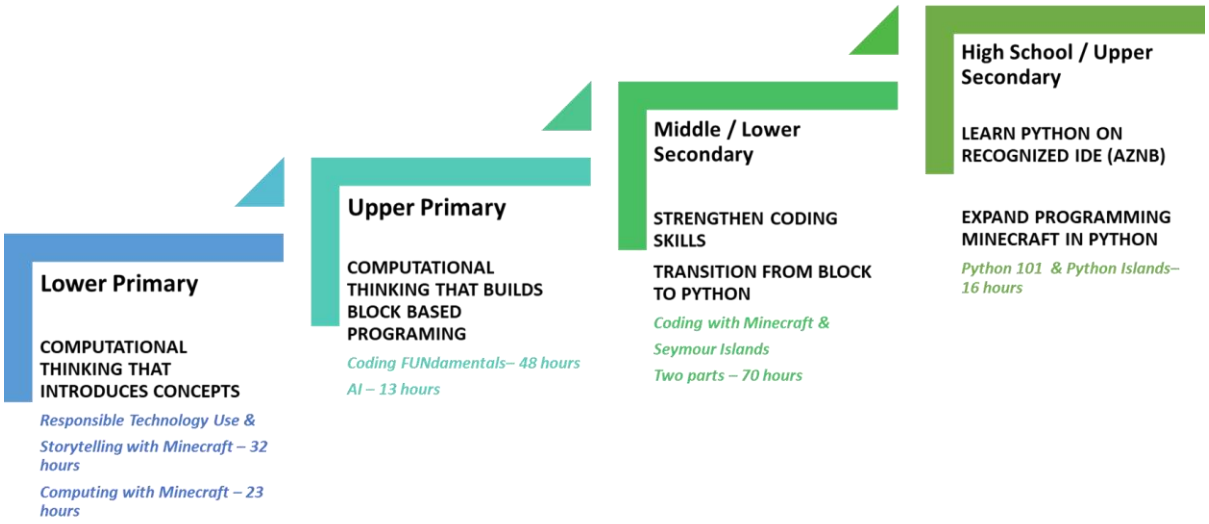
Preparing to Teach with Minecraft: Education Edition

Teachers do not need to have any prior computer science experience; however, they should familiarize themselves with a basic understanding of what is Minecraft: Education Edition. Support for building out teacher knowledge can be found here:

Minecraft: Education Edition: Teacher Academy

Minecraft: Education Edition Webinar Series

Computer Science Pathway Progression



Programming with Python is a part of the Upper Secondary (Grades 9-12|Ages 14-18) computer science progression. In this part of the progression, students are transitioning over to text-based programming utilizing Python. Python is the text-based programming language used by millions of professional coders at places like Microsoft, Google, IBM, and even NASA! Python has worldwide use across various industries and is considered one of the preferred programming languages in industry due to its stability, security, and flexibility, and straightforward syntax.

Students will begin their experience with MakeCode Python and then scaffold over to Python in Azure Notebooks. The focus within this part of the progression is to support students' transition to text-based programming and preserve the engagement in Minecraft. The goal is to build out students' knowledge in how to use Python coding to accomplish tasks and solve problems.

Curriculum Summary

Programming with Python is specially designed computer science content for students in Upper Secondary/Grades 9-12/Ages 14-18. Students will learn, practice, and apply relevant computer science skills and concepts as well as literacy skills in both unplugged and digital experiences. The lessons are designed to provide students with an opportunity to build knowledge of the concept(s) in an unplugged version (i.e., demonstrate the concept on paper), practice the concept with the direct support of their teacher in the Minecraft world, and then finally by completing the task independently.

Lesson Design

Each EDU guide contains multiple activities that are intended to be taught over the specified amount of sessions (explained in the **Instructional Sequence** part of this document); however, you should use discretion and modify/adapt the lesson activities based on your students' needs and abilities. Within the lessons, the instructional sequence will contain three parts correlating with the gradual release model:

Direct Instruction—Teacher-Directed, "I Do"	In the first step, the teacher introduces and models the appropriate way of performing the skills included in the new concept being taught.
Guided Instruction— Teacher Modeling, "We Do"	After the teacher models the correct way to understand or perform the new concept being taught, teacher will guide the students as they work through some examples together.
Independent Practice—Teacher Support, "You Do"	This step is where students demonstrate their initial level of understanding of the new concept being taught through independent practice.

Instructional Materials

Curriculum Overview	That is this document you are reading now! This will provide you with insight about the curriculum and what is taught within the curriculum.
Educator's Guides (EDU Guides)	An educator's guide is provided for each of the lessons. The guide provides a high-level overview of the lesson, learning goals, standards addressed, required preparation for the activities, the lesson plans for the activities, and any additional materials needed.
Classroom Presentations	Each unit is supported by its own PowerPoint presentation to provide structure and guide the educator through the activities for the lesson.

Formative Assessments	After each lesson in the EDU guide, there is an opportunity to check for student understanding of the concept taught within the lesson. These formative assessments are typically comprised of 2-4 questions directly related to the learning that just took place.
Summative Assessment	At the end of the entire lesson sequence, students will be provided with a performance-based task to demonstrate their new knowledge and skills learned throughout the computer science unit, Programming with Python. This performance-based task can be assessed using the provided rubric.
Minecraft World Files	The specific world files needed to experience the instructional activities have been linked directly within the Educator's Guides.

Introduction

Level: Upper Secondary | Grades 9-12 | Ages 14-18

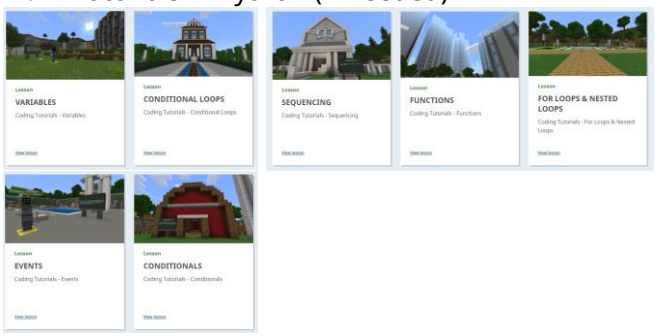
Essential Question: How can computational thinking help us to solve problems and identify solutions?

Introduction

Welcome to the text-based programming world of Python! In this unit, students will have a deep dive experience of learning the essentials of what are programming languages, what is Python syntax, and how to effectively utilize sequences, loops, conditionals, lists, and functions to develop programs. Students will be tasked with helping CodingMine to develop several types of software programs and the robot, the Agent, to help solve specific needs within the community. Students will also get to experience additional Python worlds as they travel to Python Islands to support their ongoing knowledge and skill development in Python programming.

Pacing: 18 hours (could be more if tutorials and/or supplemental lessons are included)

Materials

Hardware	• The teacher will need a laptop or tablet with a projector for the plugged-in coding lessons. • Internet access will be required for a portion of the lessons and activities. • Each student will need a device to complete the plugged-in coding activities.
Software	• Minecraft: Education Edition needs to be deployed on the devices utilized within these lessons. Use this link to find information about Deploying Minecraft: Education Edition. • The teacher and students will need the relevant MCworld files downloaded for each lesson.
Other Materials	• M:EE Tutorials in Python (if needed)  • M:EE Supplemental Lessons (if needed)

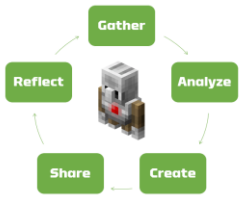
Minecraft: Education Edition Teaching and Learning Framework

<table>
<tr><th colspan="2">GOAL:
Provide students with the necessary skills to become creative coders and content creators</th></tr>
<tr><td>Coding Mindset
Although Minecraft: Education Edition provides rigorous and engaging academics, we also deliver a holistic education- providing instruction for the social, emotional, and physical needs for our students. We want to foster a distinctive set of attributes. These qualities prepare our students to make exceptional contributions both in school and outside of school.

CURIOUS
COMMITTED
COOPERATIVE
CONSIDERATE
CONFIDENT</td><td>Computational Thinking Skills
Computational thinking provides a vital skill set in which students must possess in order to fulfill the industry's needs in the jobs of tomorrow.
Our ever-changing workforce creates a critical need for innovation. Our students need computational thinking skills not just to solve the problems within their educational journey, but to also meet the challenges of adapting to our constantly changing workforce.

DECOMPOSITION
PATTERN RECOGNITION
ABSTRACTION
ALGORITHMS</td></tr>
<tr><td>Computer Science Units of Study
Minecraft: Education Edition provides meaningful, relevant, and engaging units of study. The units of study will possess a conceptual lens to allow for depth and complexity to develop conceptual understanding—knowledge which transfers through time, across cultures, and across situations.

DIGITAL CITIZENSHIP
PROGRAMMING
CYBERSECURITY
IMPACTS OF COMPUTING</td><td>Community
As students gain and possess new knowledge and skills, we strive for them to find a greater purpose of "why do I need to know this" or more importantly, "how can I use this information?".
We aim to empower students, develop confidence and self-efficacy into a commitment to serve the community in which we live in and beyond.

 
</td></tr>
</table>

This unit will focus on Programming, as students will transition to utilizing MakeCode Python and Python in Azure Notebooks. Students will learn the importance and value of text-based programming within industry.

This unit will focus on the coding mindset of being committed, as they learn new skills and concepts within Python.

This unit will focus on algorithms as students learn about the needed syntax and coding concepts to create programs with Python.

This unit will provide students with the opportunity to become a content creator within the Minecraft: Education Edition community. Students will have the opportunity to utilize the Reference Guide to build out their own game and/or computational artifact in Minecraft utilizing Python at the end of this unit.

Instructional Sequence

This next section will provide you with an overview of the activities included in this lesson sequence. The lesson sequence is presented in chronological order—we suggest working in order, as the content will build upon skills presented in the previous session. A session is equivalent to one class period, or a 45-60 minute session. However, educators should feel empowered to modify and adapt the lesson sequence to best meet the needs of their students.

Lesson Sequence Overview

Session	Objectives	Teacher will	Students will	Resources
*1-3**	*Students will engage in a multiplayer experience*	*Lead students through an unplugged activity to model and articulate the different story elements of a narrative text*	*Students will engage in a multiplayer experience*	*Seymour Island*
*4**	*Students will be able to explain what are computer languages.*	*Introduce what a computer language is, what an algorithm is, and introduce Python*	*Students will write MakeCode Python commands using correct syntax and find and correct errors in already existing pieces of code.*	*Python 101: All That Syntax*
*5**	*Students will utilize command in command structure and the code completion tool to find the most efficient way to construct objects.*	*Introduce the concept of relative coordinates and positional programming in Minecraft*	*Students will write Minecraft Python commands to place blocks in the right locations in a three-dimensional space.*	*Python 101: Location, Location, Location*
*6**	*Students will learn about and use variables and world coordinates in their Python programs.*	*Introduce how to create a variable, assign a string, assign a numerical value, and how to change the value of a variable*	*Students will help CodingMine develop a farming software (using variables and world locations) to help in every step of the farming process from planting to sorting to selling the produce.*	*Python 101: A Varied Outcome*

*5**	*Students will learn how to effectively use lists in their Python programs.*	*Introduce the concept of lists, how to create lists, how to use methods with lists, and how to modify and sort lists*	*Students will help CodingMine develop a identification software (using lists) to help veterinarians to easily identify and categorize animals as well as help them prescribe treatments and dietary requirements.*	*Python 101: Animals are Friends*
*7**	*Students will learn how to effectively use loops in their Python programs.*	*Introduce the concept of loops, the use of the "for loop", how to use nested loops, and how to use indentations.*	*Students will help CodingMine develop their top-secret project called the Agent, a robot, to do different jobs around the house.*	*Python 101: A Helper for the Home*
*8**	*Students will learn how to effectively use conditionals and Boolean logic in their Python programs.*	*Introduce the concept of conditions and Boolean logic; explain how to use if, if else and elif conditionals*	*Students will help CodingMine develop a program (using conditionals and Boolean logic) to help the Agent to drive a car by itself.*	*Python 101: Driving Around*
*9**	*Students will learn how to effectively use while loops and sequences in their Python programs.*	*Introduce the concepts of while loops and sequence; explain how to use while loops with different conditions*	*Students will develop a program (using while loops and sequences) to have the Agent respond in emergency situations by building structures like water barriers, firebreaks, and rebuilding house foundations.*	*Python 101: Emergency Response*
*10**	*Students will learn how to effectively use functions in their Python programs.*	*Introduce the concept of functions, how to use functions, how to create their own functions, and the concepts of in-code comments*	*Students will develop a program (using functions) to have the Agent help an ecological organization to prepare the soil to plant pastures of saplings.*	*Python 101: Planting A Seed*
*11**	*Students will learn how to tackle writing larger codes when utilizing all of*	*Introduce the concept of decomposition for students*	*Students write and develop one large code to create a fully working game.*	*Python 101: All Fun and Games*

	the coding concepts they have learned.			
*12**	*Students will demonstrate their ability to create Python programs.*	*Explain the summative task and provide success criteria and guidance to collaborative groups.*	*Students will utilize their accumulated coding knowledge to complete a number of challenges within a given time.*	*Python 101: Creative Coding*
*13**	*Students will learn the say() command and its purpose for outputting information.*	*Introduce the Agent and introduce the concept of inputs/outputs.*	*Students will navigate Python Island 1 to increase their coding proficiency in Python.*	*Python Island 1*
*14**	*Students will build up experience with multiple approaches to using conditional statements.*	*Review the concepts of conditional statements*	*Students will navigate Python Island 2 to increase their coding proficiency in Python.*	*Python Island 2*
*15**	*Students will build up experience with using for loops in their Python code.*	*Review the concept of for loops in Python*	*Students will navigate Python Island 3 to increase their coding proficiency in Python.*	*Python Island 3*
*16**	*Students build up experience in using while loops in their Python code.*	*Review the concept of while loops in Python*	*Students will navigate Python Island 4 to increase their coding proficiency in Python.*	*Python Island 4*
*17**	*Students build up experience in using functions in their Python code.*	*Review the concept of functions in Python*	*Students will navigate Python Island 5 to increase their coding proficiency in Python.*	*Python Island 5*
*18**	*Students build up experience in using lists in their Python code.*	*Review the concept of lists in Python*	*Students will navigate Python Island 6 to increase their coding proficiency in Python.*	*Python Island 6*
*19**	*Students will learn and use the Reference Guide for Python AZNB.*	*Introduce the Reference Guide for students' use*	*Students will utilize the Python Reference Guide to complete tasks and create their own program*	*Python Island 7*

**Teachers should use discretion and modify the lessons as needed for students based on needs.*

Educational Standards

CSTA Standards

- 3A-AP-13 Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.
- 3A-AP-14 Use lists to simplify solutions, generalizing computational problems instead of repeatedly using simple variables.
- 3A-AP-15 Justify the selection of specific control structures when trade-offs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made.
- 3A-AP-16 Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.
- 3A-AP-17 Decompose problems into smaller components through systematic analysis, using constructs such as procedures, modules, and/or objects.
- 3A-AP-18 Create artifacts by using procedures within a program, combinations of data and procedures, or independent but interrelated programs.
- 3A-AP-21 Evaluate and refine computational artifacts to make them more usable and accessible.
- 3A-AP-22 Design and develop computational artifacts working in team roles using collaborative tools.
- 3A-AP-23 Document design decisions using text, graphics, presentations, and/or demonstrations in the development of complex programs.

ISTE Standards

- 1.2.b Students engage in positive, safe, legal and ethical behavior when using technology, including social interactions online or when using networked devices.
- 1.3.c Students curate information from digital resources using a variety of tools and methods to create collections of artifacts that demonstrate meaningful connections or conclusions.
- 1.4.c Students develop, test and refine prototypes as part of a cyclical design process.
- 1.5.a Students formulate problem definitions suited for technology-assisted methods such as data analysis, abstract models and algorithmic thinking in exploring and finding solutions.
- 1.5.c Students break problems into component parts, extract key information, and develop descriptive models to understand complex systems or facilitate problem-solving.
- 1.6.b Students create original works or responsibly repurpose or remix digital resources into new creations.
- 1.7.c Students contribute constructively to project teams, assuming various roles and responsibilities to work effectively toward a common goal.

UK National Curriculum: Computing - Key Stage 4

- Develop their capability, creativity and knowledge in computer science, digital media and information technology
- develop and apply their analytic, problem-solving, design, and computational thinking skills
- understand how changes in technology affect safety, including new ways to protect their online privacy and identity, and how to identify and report a range of concerns

Australian F-10 Curriculum: Digital Technologies - Year 9 and 10

- Developing a preliminary specification for an opportunity or a need that typically contains a problem statement, a set of solution needs expressed as functional and non-functional requirements, any assumptions or constraints to be considered and the scope or boundaries of the solution (ACTDIP036)
- Designing algorithms to solve real-world problems and describing algorithms using flow charts and structured English (ACTDIP040)
- Recognising that different algorithms can solve a problem with different trade-offs (ACTDIP040)
- Tracing algorithms to predict results and program state for a given input (ACTDIP040)
- Coding separate modules that perform discrete functions but collectively meet the needs of the solution (ACTDIP041)
- Considering different algorithms and selecting the most appropriate based on the type of problem (ACTDIP041)
- Selecting different types of data structures such as an array, record and object to model structured data (ACTDIP041)
- Creating online interactive solutions for working with others by combining or modifying online software tools to support project work (ACTDIP043)
- Managing and modifying the development of a solution (ACTDIP044)
- Developing an evolutionary prototype iteratively and incrementally (ACTDIP044)

Computer Science Concepts: Programming with Python

Lesson	Concept(s)
Seymour Island	Sequencing Control Structures (nested loops and compound conditionals)
Python 101: All that Syntax	Python command syntax structure String (text) Integers (numbers) Syntax errors
Python 101: Location, Location, Location	Relative coordinates Positioning programming
Python 101: A Varied Outcome	Variables World coordinates
Python 101: Animals are Friends	Lists Zero-based numbering
Python 101: A Helper for the Home	For loops Nested loops
Python 101: Driving Around	if Conditionals if else Conditionals elif Conditionals Boolean logic
Python 101: Emergency Response	While loops Sequencing
Python 101: Planting a Seed	Functions Code comments
Python 101: All Fun and Games	Decomposition Relative positions
Python 101: Creative Coding	Collaboration Program Development Testing
Python Island 1	Using outputs (say command) Variables
Python Island 2	Conditions if statements Else statements Nested if/else statements

	Logic operators
Python Island 3	For loops Nested for loops Break points
Python Island 4	While loops Nested while loops
Python Island 5	Functions Return statements Events
Python Island 6	Lists Arrays Lists with for loops
Python Island 7	Use of Reference Guide Problem Solving Program Development Testing Collaboration

MINECRAFT VISUAL GLOSSARY

Agent

personal robot in Minecraft

Blocks

the basic units of structure in Minecraft that make up the game's world

Book & Quill

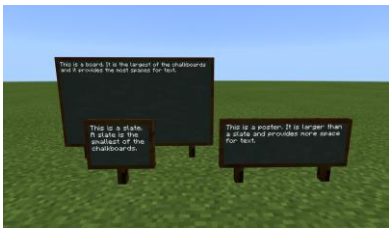
an item used to create written books in Minecraft

Camera

allows you to take screenshots and selfies in Minecraft: Education Edition

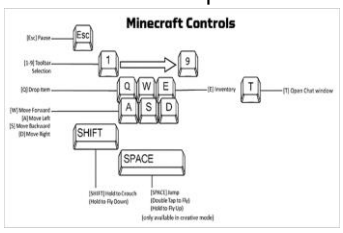
Chalkboards

special blocks that allow you to write and display text in Minecraft



Controls

(keyboard)
keyboard buttons that help you move around and complete tasks

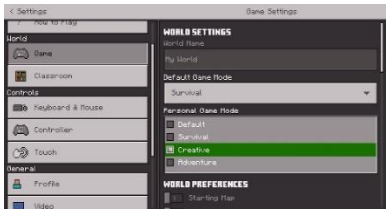


Controls

(touch)
the touch pad that helps you move around and complete tasks

Creative

game mode that gives you unlimited resources, ability to fly, and lets you destroy blocks instantly when mining



Hotbar

selection bar that appears on the bottom of the screen

Inventory

pop-up menu the player (or Agent) uses to manage the items they carry

Minecraft: Education Edition

a game-based learning platform

Mob

game character resembling a living creature

Portfolio

saves all of the photos that you have taken with a camera; allows you to add captions

NPC

non-player character

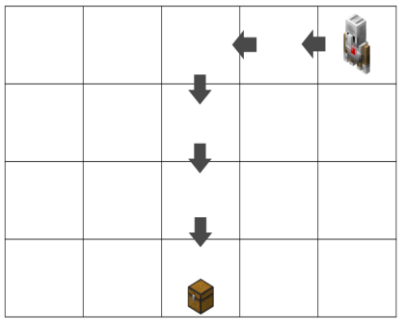
Spawn Point

the location where a player begins game play

COMPUTER SCIENCE GLOSSARY

Algorithm

a sequence of defined steps



Boolean Logic

uses the numbers 1 and 0 to represent if something is true (1) or false (0) and is called its state

1 or 0

Code (Coding)

the method of giving a computer instructions to perform a specific task

```
agent.give("dirt",64,1)
for i in range(2):
    agent.move("forward")
    agent.place(1, "down")
```

Colon

a symbol used at the end of command declaration

:

:

Commas

used to separate parameters if there are two or more

```
command_group.command_name(parameter1, parameter2, parameter3)
```

Comments

pieces of a text in a code that the computer does not run

```
# This function makes the Agent turn right
def turn_right():
    agent.turn(RIGHT_TURN)
    agent.move(FORWARD, 1)
    agent.turn(RIGHT_TURN)
```

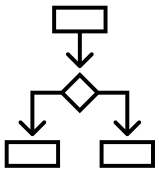
Compare

double equals, or compare, is used when comparing two statements

==

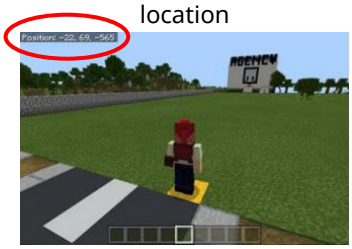
Conditionals

an action or occurrence that is detected by the computer



Coordinates

a coordinate represents a position or location

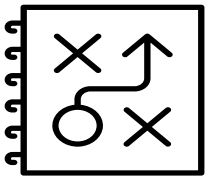


Debug

the process of finding (and correcting) errors in a computer program

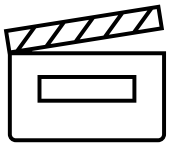
Decompose

breaking down a problem or task into smaller, more manageable parts



Event

an action or occurrence that is detected by the computer



Function

a named section of a program that performs a specific task

```
function_name():
```

Indentation

declare the piece of code belongs to the command above; to indent a piece of code, use the tab key on the keyboard

```
for i in range(2):
    agent.move(FORWARD, 1)
```

List

An ordered collection of strings, numerical values, or variables

```
animal_list = [elephant, lion, zebra, giraffe]
```

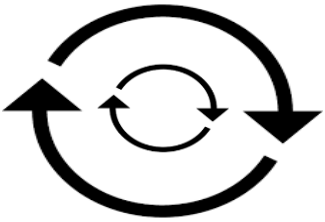
Methods

special commands that affect data in many ways

`my_list.sort()` – Sorts a list in alphabetical order.
`my_list.reverse()` – Reverses the order of the contents.
`my_list.append()` – Adds another value to the end of a list.
`my_list.pop()` – Removes a value from a list.

Nested Loops

a loop within a loop



Parentheses

used after a command name; can have parameters placed in them to set the command's settings, to make the command do exactly what is wanted

Python

a computer programming language that uses a mixture of recognizable words and characters

```
def on_chat():
    for i in range(100):
        mobs.spawn(CHICKEN, pos(0, 10, 0))
player.on_chat("chicken", on_chat)
```

Quotes

used to define a string; any characters between the quotes will be seen by Python as text

Sequence

a programmed (coded) algorithm

Square Brackets

used when creating lists; contents of a list need to be placed within square brackets and separated by commas

```
name_list = [John, Mary, Chad, Isabella]
```

String

a piece of text in quotes "-"

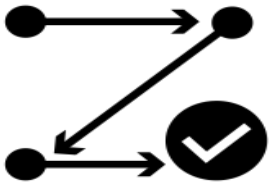
"Hello"

Syntax Error

the computer's way to say that it cannot run a piece of code because the syntax structure is not correct

Test

the process of running a program to see if it is correct



Underscore

used in the place of a space

```
my_list = [0, 1, 2, 3, 4]
important_variable = 1
```

Variable

the places in which computers store values in a computer program

```
# Say Hello!
greeting = "Hello World!"
say(greeting)
# Introduce yourself!
myname = "Anonymous"
say(myname)
# try both
say(greeting,myname)
```

While Loops

repeats code when a condition is met

World Position

unique coordinates defined by three numbers

```
world(0, 0, 0)
```