Introduction to Computational Thinking and Python Programming

Instructor: Sarom Leang, PhD (Teacher/Professor)

Assistant: Romin Katre

Innovation Hall 203

Pronouns

- Instructor: Sarom Leang, Ph.D. (Teacher/Professor/Instructor)
- Assistant: Romin Katre

Schedule

- 10:00 AM – 10:15 AM (Homeroom)
- 10:15 AM – 11:35 AM (G1)
  - 10:45 AM – Code Combat
  - 11:20 AM – Exit Survey
- 11:40 AM – 12:35 PM (Lunch)
- 12:40 PM – 02:00 PM (G2)
  - 1:05 PM – Code Combat
  - 1:40 PM Exit Survey
  - 1:50 PM Student Survey

Student Expectations

- **NO FOOD**
- **NO DRINKS** (on the table)
- Be respectful to individuals and property
- Be open to learning
- Be open to not understanding
- Be patient with yourself
- Ask questions
- Explore
- **Embrace failure**

G1

- If we start class 5 minutes early (e.g., at 10:10 AM), then the class will be dismissed to lunch 5 minutes early at 11:30 AM.

G2

- If you arrive in your seat by 12:40 PM then you have the option to finish your lunch in the hallway – if needed.

Course Overview

- This course introduces the fundamental building blocks of computational thinking and computer programming using the Python language.
- Upon successful completion of this course, students will be able to:
  - Improve their problem-solving skills
  - Write, read, and execute Python code using basic data types and operators

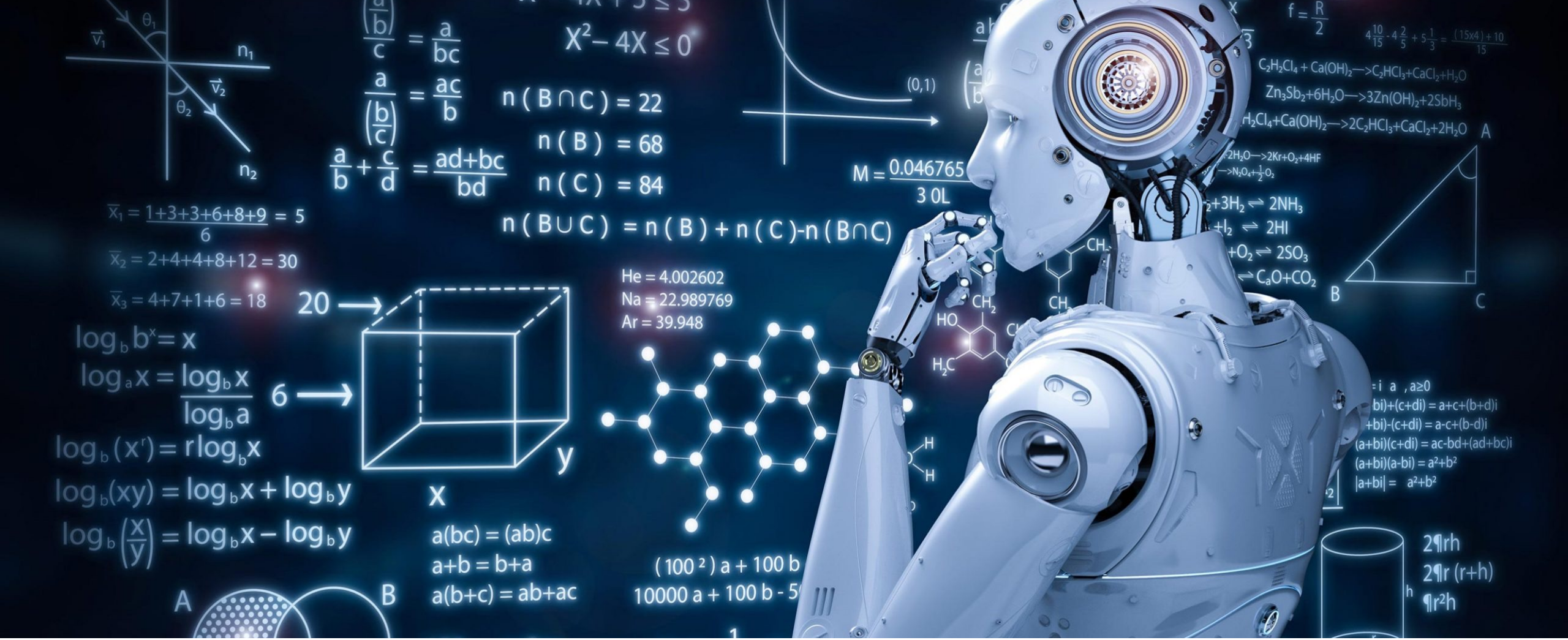
Course Overview (cont.)

Exploratory Topics

- ~~Session #1 – October 23, 2023~~
  - ~~Entrance Survey~~
  - ~~How Computers Work~~
- ~~Session #2 – December 2, 2023~~
  - ~~Parallel Computing~~
- ~~Session #3 – February 10, 2024~~
- ~~Session #4 – March 23, 2024~~
  - ~~Open Lab~~
- Session #5 – April 20, 2024
  - Generative Artificial Intelligence
  - Exit survey

Topics

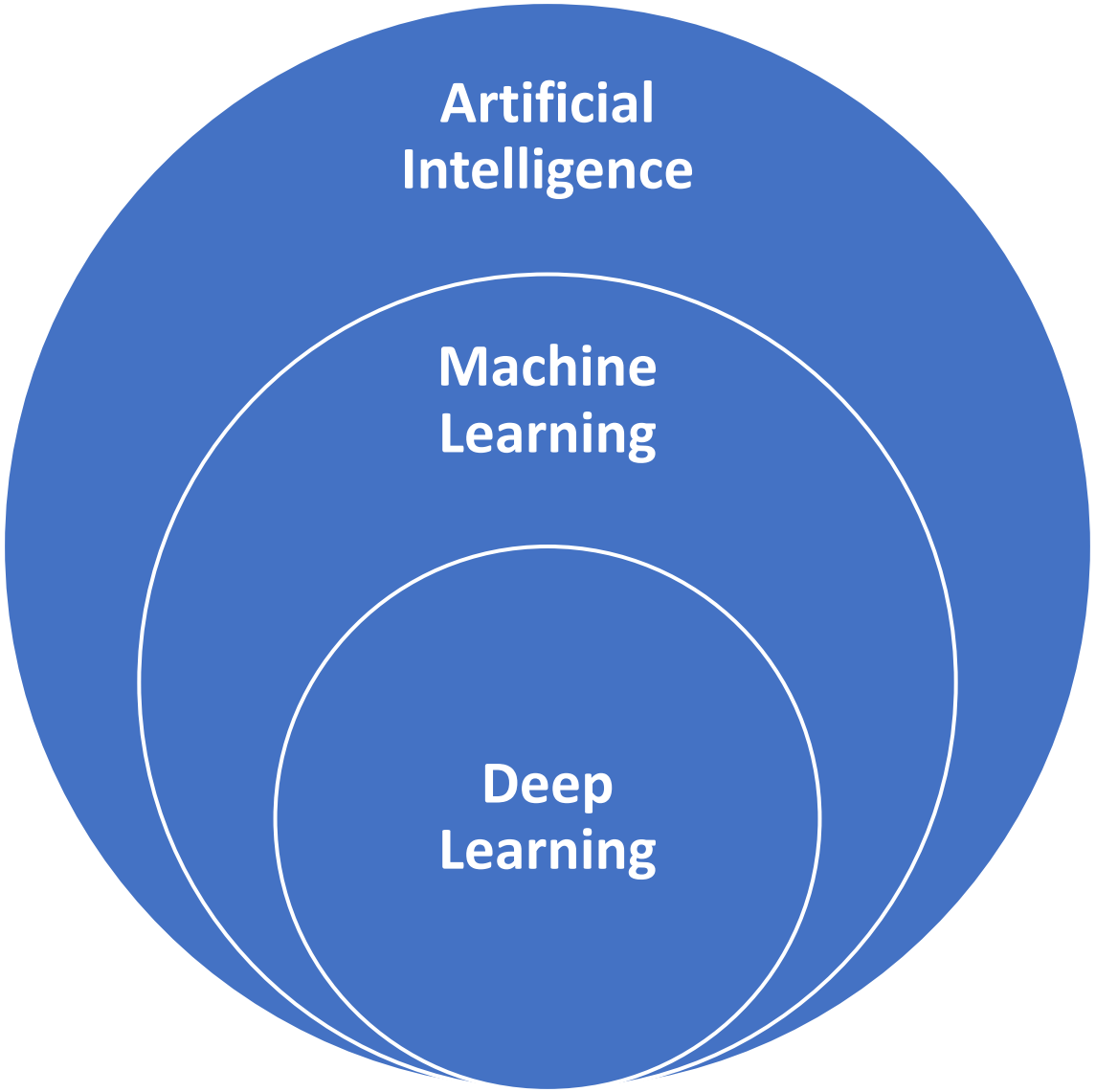
- Algorithms and Problem Solving
- Debugging and Troubleshooting
- Loops
- Algorithms and Syntax
- Variables and Conditionals
- Variable Arithmetic
- Conditionals (If/Else)
- Compound Conditionals



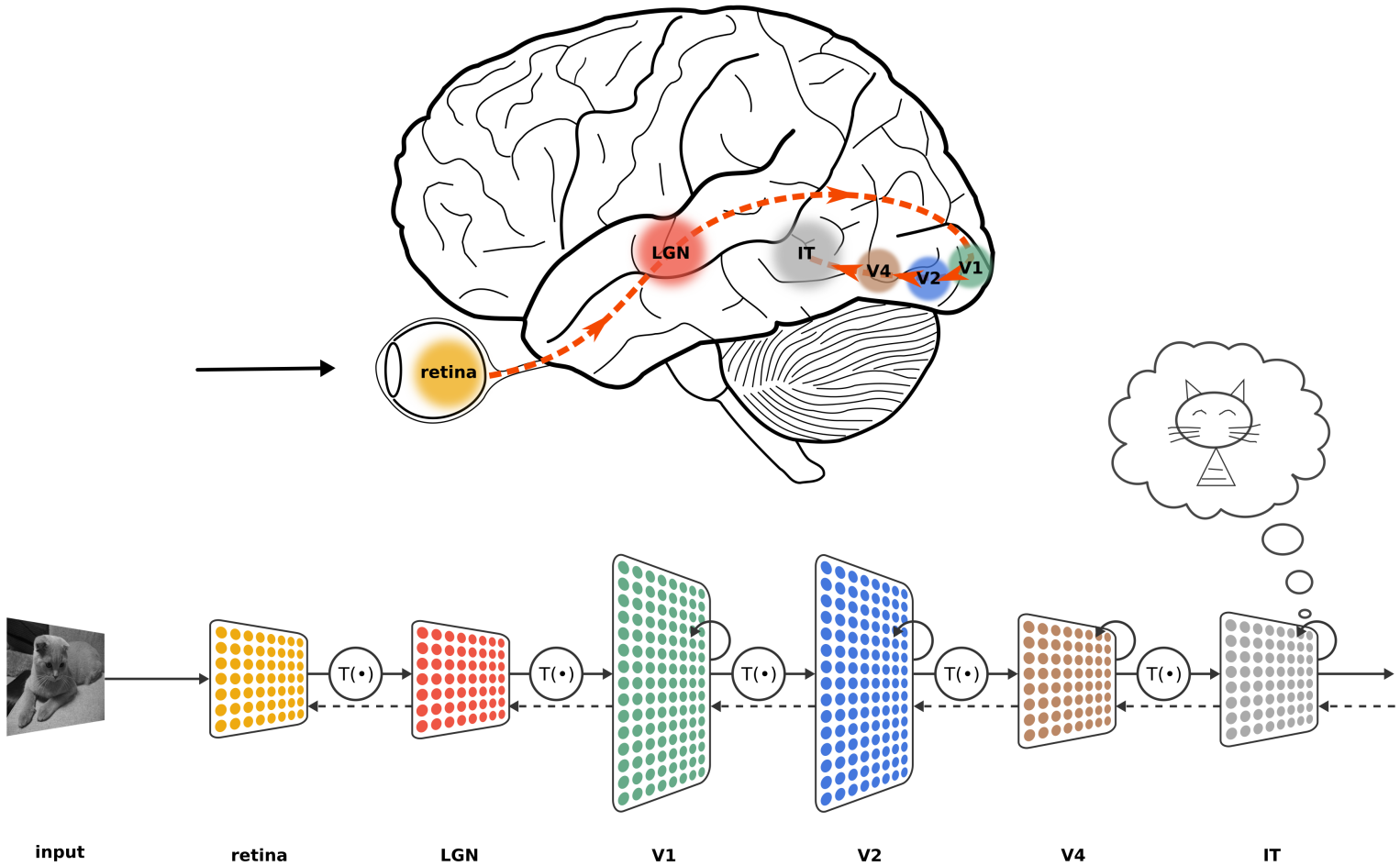
Generative AI

Buzzwords: AI/ML/DL

- AI - Creation of **intelligent machines** that can **perform tasks** that would typically **require human intelligence** to complete.
- ML - teaching computers to **learn** and **make decisions** on their own.
- DL - type of ML that uses multi-layer neural networks to **analyze** and **interpret data** (**recognize patterns**).



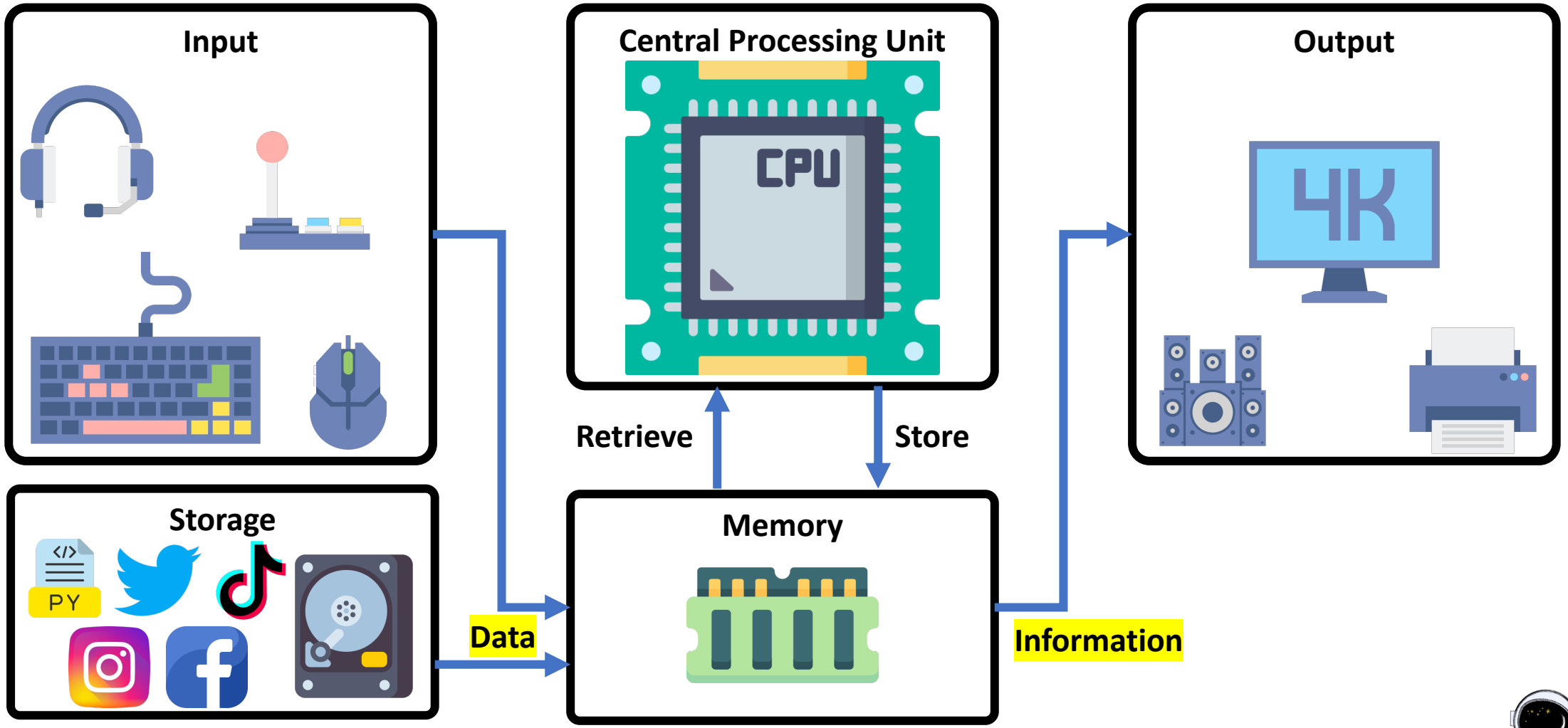
Neural Networks



What is Generative AI?

- Generative AI is a type of artificial intelligence that can **create new content that is similar to the original data it was trained on**.
- It **learns patterns and relationships in data**, and **uses that knowledge to generate new content**, such as images, music, or text.
- It's used in many different applications, including art, music, writing, and video game development.
- It can also be used to create more realistic simulations for scientific research and to **generate synthetic data** for machine learning models.

Data -> Information



Data -> Information -> Knowledge

Data (raw data)	Information (processed and organized)	Knowledge (insight gained)
Census records	Demographics	What will be the average age of the population in 2040?
Weather measurements	Average temperature, rainfall, etc. in a region over a given time range	How will the weather be next week, month, year?
Financial records	Growth rate of a company	Where will this company be in the next five years?
Patient records	Incident rate of a disease	When will COVID-19 end?

Popular Generative AI Tools

- **ChatGPT**: This is a language model that can generate human-like text based on a given prompt.
- **Microsoft 365 Copilot**: AI for Microsoft Office
- **Midjourney/DALL-E**: This is an image generation AI that can generate images from textual descriptions.
- **MuseNet**: This model can generate original music along with multi-instrumental pieces.

ChatGPT

- Interact with this AI model by providing prompts.
- Prompts may describe a task being asked of the model.
- Prompts are inputs to the model.
- Generative content are the outputs from the model.
- The performance (quality of the output) of these models are dependent on the inputs.
- Prompt Engineering - creating specific prompts or inputs that can guide the generative AI model to produce output that is relevant and coherent.

ChatGPT Prompts

- Classify the named entities in this text: George Washington and his troops crossed the Delaware River on December 25, 1776 during the American Revolutionary War.
- Translate this text into Portuguese: Welcome to the Matrix
- Explain this python code:

```
r=int(input("Enter upper limit: "))
for a in range(2,r+1):
    k=0
    for i in range(2,a//2+1):
        if(a%i==0):
            k=k+1
    if(k<=0):
        print(a)
```

ChatGPT Prompts

- Write a positive Yelp review of Café Renaissance in Vienna, Virginia. Highlighting their grilled halibut dish in almond and lemon butter sauce.
  - More witty and colorful.
  - End with: It was a wonderful way to spend Valentines Day.
  - Also highlight the Escargot à la Bourguignonne appetizer which was cooked in garlic butter parsley and brandy.
- Create a template for a letter of recommendation for Dr. Sarom Leang. Dr. Leang was an Early Identification Program scholar during undergrad. Dr. Leang earned his BS in chemistry and a minor in computer science from George Mason University in 2004 and completed his PhD in quantum chemistry from Iowa State University in 2012.

ig.ft.com/generative-ai

How Do Large Language Models (LLMs) Work?

- LLMs are statistical pattern-recognition and prediction systems
- LLMs output the next likely word (“token”) in a sentence (“sequence”)
  - token: unit of text e.g. word, character. 1 word ~ 0.75 token
  - sequence: context - section (“window”) of text e.g. sentence, paragraph, book
  - input into chatGPT is 4096 tokens; Claude 2 is 100K tokens
- The likelihood of the next work appearing is determined by
  - the context in which the words are seen in a larger body of text (“corpus”) and
  - the input to the chat

DALL-E: An abstract image of George Mason on a horse riding into a rainbow sunset




DALL-E: A painting of Michael Jordan slam dunking in the air while eating an avocado

DALL-E: Synthwave George Mason University logo

DALL-E: A photo of an avocado playing basketball with a pomegranate




DALL-E: A painting of dark-skinned female Santa having dinner with dark-skinned female Jesus

Generative AI Inception

- Ask ChatGPT to generate a prompt for DALL-E.

Create an image of a fantastical underwater city, with towering structures and bustling streets, where mermaids and other sea creatures live and work alongside human-like beings. The city should have a distinct aesthetic, with colors and architectural elements that evoke the feeling of being submerged in the depths of the ocean. The image should also include unique transportation methods, such as aquatic vehicles or creatures that are used for travel.

Generative AI Limitations

- Generative AI can produce content that is **incorrect** or **incoherent** (hallucinate).
- Generative models don't understand the content they generate. They **generate responses based on statistical patterns rather than comprehension**, which means they can't answer questions that require understanding or reasoning.
- Generative AI can perpetuate **biases** present in the training data.
- Generative models can be **verbose** and tend to **over generate** content.
- Same prompt may produce **inconsistent** responses using the same model.

Generative AI and Academic Code of Conduct

- Cheating or violating academic code of conduct using generative AI is unethical and has serious consequences.
- Disciplinary action may include failing the assignment or course, or even expulsion from school.
- Long-term consequences may include damaging a student's reputation and future academic and career prospects.
- **Use generative AI responsibly and in accordance with the academic code of conduct.**

Ozaria and Code Combat

G1

- Go to www.ozaria.com and sign-in
- Chapter 1 (Introduction to Coding)
  - 4 out 20 students have completed all modules ***now 12 out of 20***
- Chapter 2 (Algorithms & Syntax, Debugging, Variables, Conditionals)
  - 2 out of 20 students have completed all modules ***now 4 out of 20***
- Chapter 3 (Review, For Loops, Nesting, While Loops)
  - 2 out of 20 students have completed all modules ***still 2 out of 20***
- Chapter 4 (Compound Conditionals, Functions and Data Analysis)
  - 2 out of 20 students have completed all modules ***still 2 out of 20***

G1: CodeCombat

- Visit **codecombat.com**
- Click on **I'm a Student**
- Use Class Code: **SnowRunCorn**

G2

- Go to www.ozaria.com and sign-in
- Chapter 1 (Introduction to Coding)
  - 1 out 17 students have completed all modules ***now 8 out of 17***
- Chapter 2 (Algorithms & Syntax, Debugging, Variables, Conditionals)
  - 0 out of 17 students have completed all modules ***now 1 out of 17***
- Chapter 3 (Review, For Loops, Nesting, While Loops)
  - 0 out of 17 students have completed all modules ***still 0 out of 17***
- Chapter 4 (Compound Conditionals, Functions and Data Analysis)
  - 0 out of 17 students have completed all modules ***still 0 out of 17***

G2: CodeCombat

- Visit **codecombat.com**
- Click on **I'm a Student**
- Use Class Code: **StarGoodGift**

Your Adventure Awaits...

Our Story Begins in the Dungeon…

…where you will complete a series of quests to survive, work with your allies, and escape! Along the way you will:

Learn the **basic syntax** needed to find your way through dungeon mazes, collect gems, and avoid danger!

Unlock items to alter your hero's **properties** to make them survive longer, hit harder, and move faster!

Use **while loops** to repeat blocks of code!

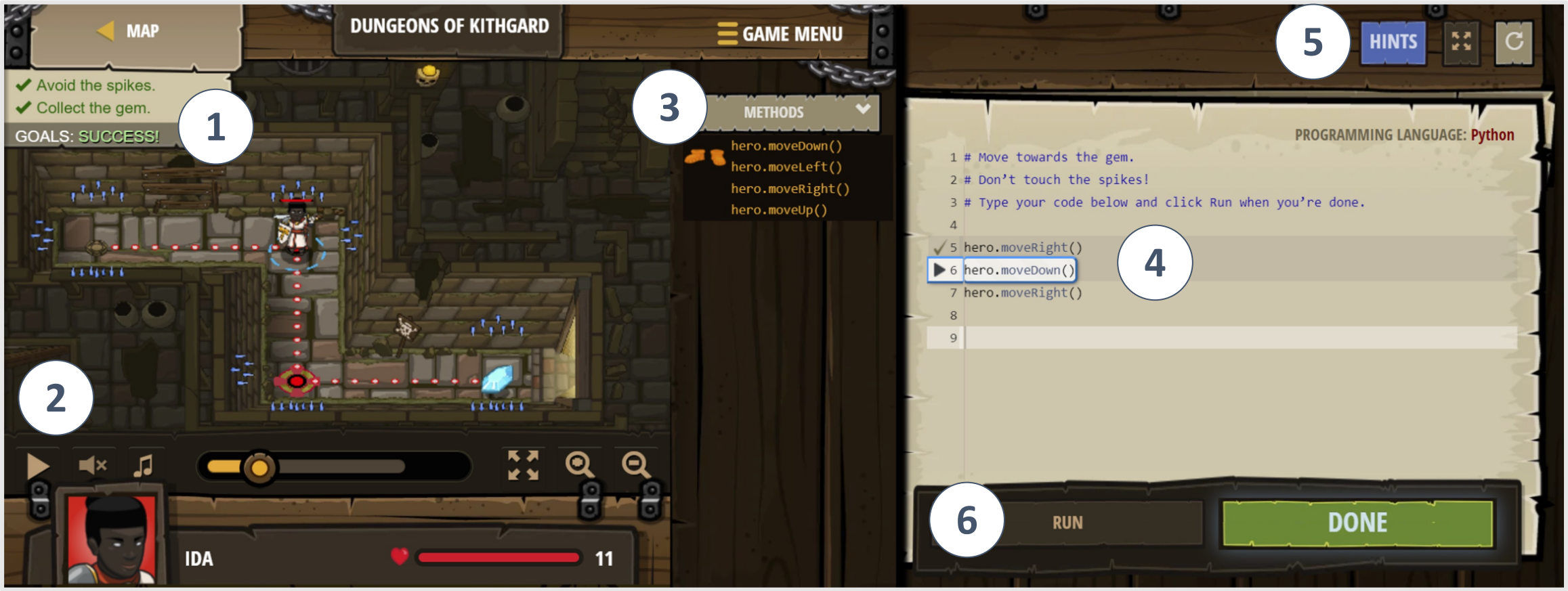
Assign values to **variables** to keep track of your enemies!

Complete a **CAPSTONE PROJECT** where you will learn to solve problems with the Engineering Design Process!

CODE COMBAT

STEM FUSION!

Overview of CodeCombat IDE



1. Level goals and completion status
2. Game controls (play/pause, volume, clock slider, zoom) and select object properties
3. Statements and code segments available in the level; click on any method to learn how to use it
4. The editor window is where you will write your own code. There is an autocomplete feature!
5. Click "**HINTS**" for more information if you get stuck.
6. Click "**RUN**" to test your code. Test often!

Syntax - Calling Methods

This statement **calls** a method! A method call contains three pieces of information:

1. **object:** The hero.
2. **method:** The action the object performs.
3. **argument:** A value that describes how many steps the hero takes to the right.

Python Syntax:

```
# Comments will include
# instructions and hints!
hero.moveRight()
```

object

method

optional arguments

Program Documentation

```
# Document your programs by
# adding comments to help
# maintain, extend, and
# improve upon your
# programs!

hero.moveRight()
```

Documentation: The written descriptions we include within our programs to explain how code segments work. We often accomplish this by writing **comments** in our programs.

It is important to document your programs all the time and at multiple stages of the development process!

Finding and Fixing Errors

You're going to make mistakes. That's a very NORMAL part of coding, even for experts. Finding and fixing errors (**debugging**) is an important part of programming. Consider using the following tools when debugging your programs:

- **Error messages**
- **Comments**
- **`hero.say()` or `print()`**
- **Inputs and outputs**
- **Tracing tools**
- **Debuggers**
- **Unit tests**

We are going to practice reading error messages in CS1!

CODE COMBAT

Method Arguments

An **argument** contains information that describes how an action will be performed. We **pass** arguments to method calls.

What homework should I do?

These heroes have gathered to do homework but they aren't sure what homework to do or for how long.
What values could you give our heroes that would help them do their homework?
Imagine you're calling the method named **doHomework**, *like so:*

How long should I work?

hero.doHomework(?)

CODE COMBAT

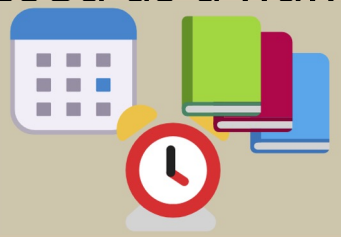
STEM FUSION

Types of Values

Computers differentiate between information that represents a ***quantity*** and information that represents ***text***.

NUMBERS

Which of your examples could be expressed as a number?



```
hero.doHomework( ? )
hero.doHomework( ? )
```

TEXT

Which of your examples could be expressed as a word or a string of

```
hero.doHomework( ? )
hero.doHomework( ? )
```

Numeric Data Types

There are two different **types** of numbers that you will want to use in your Python programs.

*We should use both **integers** and **floats** in our Python programs!*

CODE COMBAT

-4

-2

100

2663

Integers

(Whole Numbers)

2.3

36.33

-1.8

-0.5574

Floats

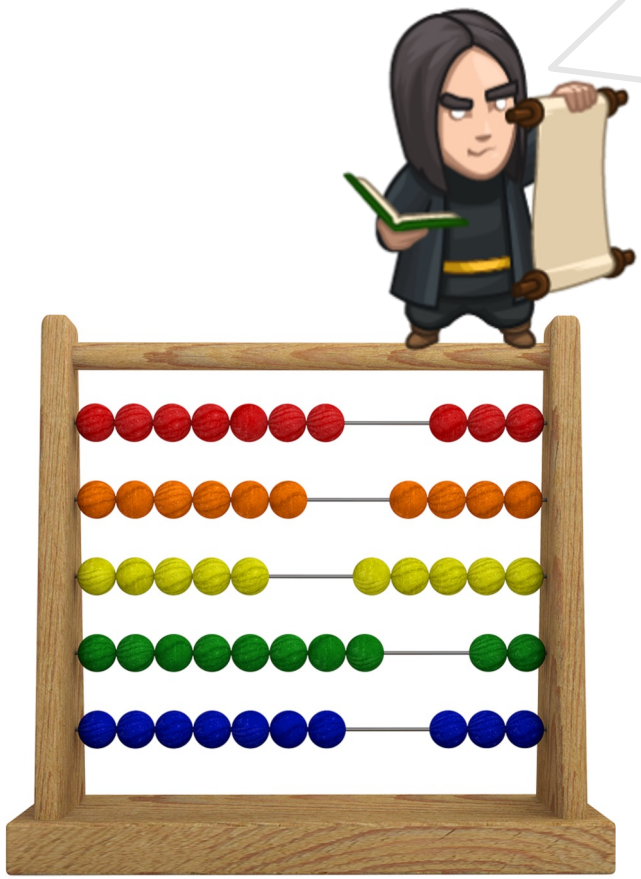
(Decimals)

STEM FUSION

Integers as Arguments

The method **countRed** defines the counting action to be performed with the abacus. The method's argument describes how many beads to count as part of the action.

The statement **abacus.countRed(3)** will count 3 red beads and move them to the right.




```
abacus.countRed(3)
abacus.countOrange( ? )
abacus.countYellow( ? )
abacus.countGreen( ? )
abacus.countBlue( ? )
```

CODE COMBAT

STEM FUSION

Text Data Type

A **string** is a sequence of characters, often used to represent text (e.g. words, sentences, etc.).

The sequence of characters you write must be contained within quotation marks if you are creating a string.

A **substring** is a smaller part of a string.

Examples of strings:

“ “

“hello world”

“CodeCombat is awesome!”

Examples of substrings:

“Charm” → “arm”

“clever” → “lever”

“Language” → “age”

CODE COMBAT

Strings as Arguments

The method **cook** defines the action of cooking a meal. The method's argument describes the meal prepared, like "Spaghetti" or "Pizza".

Remember, a string must always be enclosed in quotation marks.

Which of the following code statements will successfully help Omarn make a delicious snack for his friends?

Object: chef

Method: cook

Argument Type: string

a) **chef.cook.Gyro**
b) **chef.cook("tacos")**
c) **chef.cook(Popcorn)**
d) **chef.cook("Gryo")**

"Popcorn"

"Gyro"

"Taco"

Multiple Arguments

Sometimes, a method will need multiple arguments in order to work.

Separate each argument with a comma.

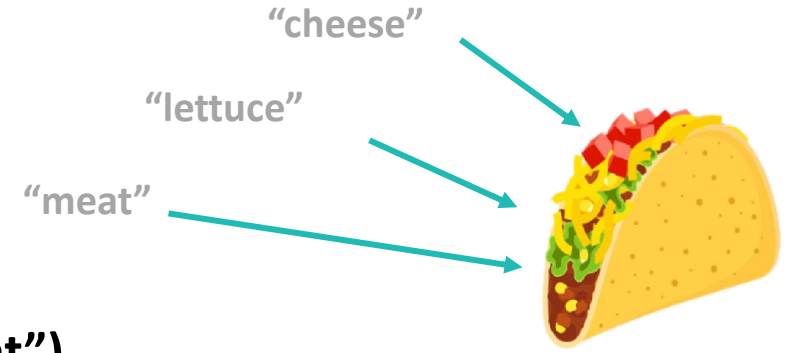
Which of the following code statements will successfully help Omarn make tacos his friends?

Object: chef

Method: cook

Argument Types: strings

a) **chef.cook.makeTacos()**
b) **chef.makeTacos(lettuce, cheese)**
c) **chef.makeTacos(“lettuce”, “cheese”, “meat”)**
d) **chef.makeTacos(“lettuce” “cheese” “meat”)**



Properties

Characteristics that all **objects of a single type** share.

For example, all hero objects have a **name** property. The values for each object's name may differ.

Boolean

A data type that represents **true** and **false**. That's it!

Python:

```
hero.say(True)
hero.say(False)
```

Boolean values are not strings.
That is, they are not enclosed by quotation marks.

While Loops

Loops are a way of repeating a sequence of code *while* a **condition** is *true*.

For example, you use an umbrella *while* it is raining. This is how we would translate this example into **pseudocode**.

It's raining! Where's my umbrella?

Keyword **Condition**

while it is raining

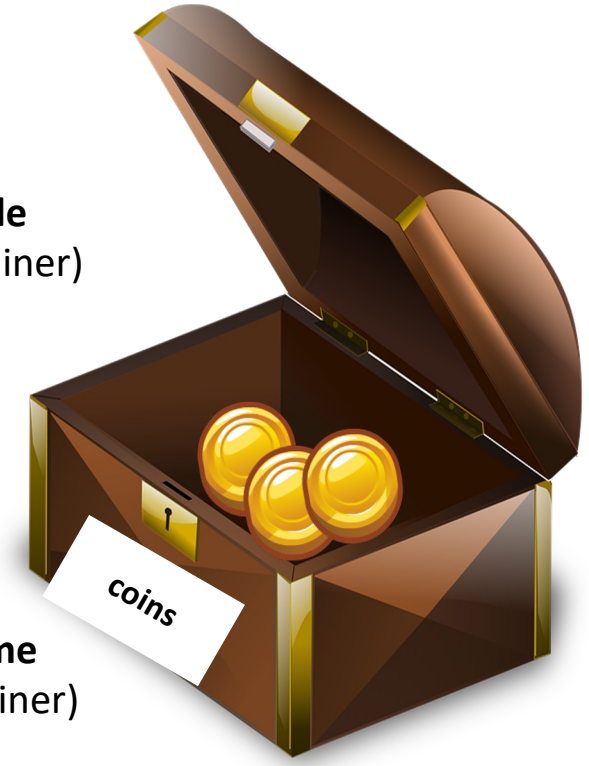
use umbrella } **Statements to repeat**

Variable

A **variable** allows us to save information for later use in our programs! The values saved in a variable can change over time.

In order to save a value in a variable, you must **assign** the value to a unique name using the **assignment operator** (=).

The integer 3 is assigned to the variable named **coins**.



Variable
(the container)

Value
(stored inside the container)

Variable Name
(name of container)

coins = 3

From Pseudocode to Syntax

Python Syntax:

name = value

You choose

Float, Integer, Boolean, String

Assignment Operator

Variable names cannot start with numbers.

Variable names cannot contain spaces or special characters!

Reassignment

You can change the value stored in a variable through **reassignment**.

The most recent assignment is the value stored in the variable – all previous values have been forgotten!

1. name = “Tharin”
2. hair = “Brown”
3. hasArmor = False
4. name = “Tharin the Knight”
5. hair = “Blue”
6. hasArmor = True

CODE COMBAT

STEM FUSION

Referencing Variables

Sometimes you will need to use the value stored in a variable somewhere in your program. In these situations, you need to **reference** the variable. To do this, write the name of the variable where you need the value.

```
1. address = "12 Cabbage Lane"
2. hero.goTo(address)
3. address = "784 Pixie Place"
4. hero.goTo(address)
```

Assigning to Method Calls

Some methods **return** information when called. In other words, the method gives back a value. You can assign that value to a variable.

```
enemy = hero.findNearestEnemy()
```

2

*The closest enemy is **ASSIGNED** to the variable* **enemy**.

1

*This method finds the enemy that is closest to the hero and **RETURNS** that enemy.*

Ursa

Brak

CODE COMBAT

STEM FUSION

Assigning to Method Calls

Here is another example of a method that **returns** information when called.

```
distance = hero.distanceTo("Ursa")
```

2

*A float is **ASSIGNED** to the variable* **distance**.

1

*This method **RETURNS** the distance between the hero and the object named "Ursa".*

Ursa

5.3 Feet

CODE COMBAT

STEM FUSION

Course Exit Survey

bit.ly/Exit24-G1

Course Exit Survey

bit.ly/Exit24-G2

STEM Fusion Survey

Please take this survey to let us know your experience in STEM Fusion!