**Lesson 1 – What is Optimization? Why Python?**

# Introduction

Imagine you wake up a little late. Do you walk, bike, or take the bus to get to school on time? Choosing the option that gets you there fastest is an example of **optimization**. In general, optimization means finding the *best solution* to a problem, saving time, effort, or resources while still reaching your goal.

- **Optimization** helps us make smart choices every day:
  - Choosing the fastest route to school
  - Packing your backpack so everything fits
  - Balancing study time and fun
  - Getting the most pizza toppings under a budget

*Packing*

*Decision Making*

*Time Management*

## Why Python for Optimization?

Python is like LEGO for coding: simple pieces that snap together to build powerful creations.

- Easy to learn: the code looks close to plain English.
- Powerful & popular: used in science, engineering, data, and apps you know.
- Free! anyone can install it and start experimenting.

We will use Python to model problems, try ideas quickly, and see results right away. Because Python is an open-source language, there are many existing optimization libraries available, so we often don't need to write algorithms from scratch.

## Getting Started with Python

Before we write our first program, we need a place to run Python code. There are two easy options:

**Option 1: Google Colab**

- Go to colab.research.google.com.
- Sign in with your Google account.
- Click on **New Notebook**.
- Type Python code in a cell and press `Shift + Enter` to run it.

**Option 2: Install Python on Your Computer**

- Visit python.org/downloads.
- Download the latest version of Python for your system (Windows, macOS, or Linux).
- Run the installer. On Windows, make sure to check the box **Add Python to PATH**.
- After installation, open **IDLE** (Python's editor).
- In IDLE, type your code and press **Run** → **Run Module** (or `F5`).

For this course, you may use either option. Google Colab is easiest to get started, but installing Python gives you more flexibility.

# Variables in Python

**What is a variable?**

A **variable** is like a little box with a label. You store information inside and use it later. In Python, the `=` sign *assigns* a value to a variable.

**Example: Adding fruits**

```
apples = 5
bananas = 3
total = apples + bananas
print(total)
```

*Output*

```
8
```

**Numbers and Text ("strings")**

- **Numbers** can be added, subtracted, multiplied, divided, etc.
- **Strings** are text wrapped in quotes.

```
name = "Jordan"
age = 16
```

```
print("Hello", name)
print("You are", age, "years old.")
```

*Output*

```
Hello, Jordan
You are 16 years old.
```

### Common Math Operations in Python

```
a = 12
b = 5

print("a + b =", a + b) # addition
print("a - b =", a - b) # subtraction
print("a * b =", a * b) # multiplication
print("a / b =", a / b) # division (keeps decimals)
print("a // b =", a // b) # floor division (rounds down)
print("a % b =", a % b) # remainder (modulus)
print("a ** 2 =", a ** 2) # exponent (a squared)
```

*Output*

```
a + b = 17
a - b = 7
a * b = 60
a / b = 2.4
a // b = 2
a % b = 2
a ** 2 = 144
```

### Mini Optimization: Buying Candy

You have $10 and each candy costs $3. How many candies can you buy?

```
money = 10
price = 3
candies = money // price # '//' divides and rounds down
print("You can buy", candies, "candies")
```

*Output*

```
You can buy 3 candies
```

**Lesson 1 Try Practice Exercises**

1. Create two variables for your favorite snacks (e.g., `cookies`, `chips`) and print their total.
2. Store your name in a variable and print a greeting.
3. If you have $20 and a video game costs $7, print how many games you can buy.
4. Compute the total cost of 3 notebooks at $2.50 each, including a 6% tax.