**Minor in AI**

**Building Blocks of Python - Revision**

19 March 2025

Introduction to programming and its relevance in AI/ML

- Variables: Definition, declaration, initialization

- Data types:

- Primitive types (int, float, str, bool), type()

- Typecasting and type conversion - int(), str(), bool()

- Arithmetic operations on int and float: +, -, \*, /, \*\*, //

- String concatenation

Alien Case Study

m1 = 73

m2 = 80

m3 = 76

print(bool(m1))

print(float(m1))

new\_m1 = chr(m1)

new\_m2 = chr(m2)

new\_m3 = chr(m3)

print(new\_m1, new\_m2, new\_m3)

s = new\_m1 + new\_m2 + new\_m3

print(s)

Conversions

# Original values

num\_str = "100"

float\_num = 45.67

zero = 0

non\_empty\_string = "Hello"

# 1. str() - Convert to string

print("Using str():")

print(str(123))               # int to str → '123'

print(str(45.67))             # float to str → '45.67'

print(str(True))              # bool to str → 'True'

print()

# 2. int() - Convert to integer

print("Using int():")

print(int(num\_str))           # str to int → 100

print(int(float\_num))         # float to int (truncates)

print(int(True))              # bool to int → 1

print(int(False))             # bool to int → 0

print()

# 3. bool() - Convert to boolean

print("Using bool():")

print(bool(zero))             # int 0 to bool → False

print(bool(10))               # int 10 to bool → True

print(bool(""))               # empty string to bool → False

print(bool(non\_empty\_string)) # non-empty string to bool → T

print()

# Additional Example: Typecasting during arithmetic

print("Arithmetic Example with Typecasting:")

result = int("20") + 5

print("int('20') + 5 =", result)

# Arithmetic Operations on int and float

# Declare int and float variables

a = 10          # int

b = 3           # int

x = 5.5         # float

y = 2.0         # float

print("Arithmetic Operations on int:")

print(a, "+", b, "=", a + b)       # Addition

print(a, "-", b, "=", a - b)       # Subtraction

print(a, "\*", b, "=", a \* b)       # Multiplication

print(a, "/", b, "=", a / b)       # Division (float result)

print(a, "\*\*", b, "=", a \*\* b)     # Exponentiation

print(a, "//", b, "=", a // b)     # Floor Division

print()

print("Arithmetic Operations on float:")

print(x, "+", y, "=", x + y)       # Addition

print(x, "-", y, "=", x - y)       # Subtraction

print(x, "\*", y, "=", x \* y)       # Multiplication

print(x, "/", y, "=", x / y)       # Division

print(x, "\*\*", y, "=", x \*\* y)     # Exponentiation

print(x, "//", y, "=", x // y)     # Floor Division

print()

print("Mixed int and float operations:")

print(a, "+", x, "=", a + x)       # int + float

print(b, "\*", y, "=", b \* y)       # int \* float

print(a, "/", y, "=", a / y)       # int / float

print(x, "\*\*", b, "=", x \*\* b)     # float \*\* int

# Define strings

str1 = "Hello"

str2 = "World"

# Method 1: Using + operator

result1 = str1 + " " + str2

print("Using + operator:", result1)

# Method 2: Using join()

result2 = " ".join([str1, str2])

print("Using join():", result2)

# Method 3: Using \* operator for repetition

repeat\_str = str1 \* 3

print("Using \* operator for repetition:", repeat\_str)

# Method 4: Concatenating with numbers (after type conversion)

num = 2025

result3 = str1 + " " + str(num)

print("Concatenating string and number:", result3)