Week 3 materials

topics covered

- numbers
- arithmetic operations
- operator precedence
- more on strings

some recap

general:

- use `type()` function to determine the data type

integers:

- integer is a whole number with no decimal places
- name for the integer data type is `int`
- to convert a string into an int, just use function `int()` e.g. `int("25")`

- in Python, you can't use commas to group digits in integer literals, but you can use underscores (`_`): 1000000 and
  1_000_000 are valid ways to represent the number one million
- there's no limit to how large an integer can be

float:

- floating-point number, or float for short, is a number with a decimal place
- to convert a string into a float, just use function `float()` e.g. `float("1.25)`
- there are three ways to represent a floating-point literal. Each of the following creates a floating-point literal with a value of one million: `1000000.0`, `1_000_000.0`, `1e6`
- third approach uses **E notation**(exponential notation) to create a float literal

  - To write a float literal in E notation, type a number followed by the letter `e` and then another number. Python takes the number to the left of the `e` and multiplies it by `10` raised to the power of the number after the `e`. So `1e6` is equivalent to 1×10^6

arithmetic operations:

addition:

- addition is performed with the `+` operator: `1 + 2`
- two numbers on either side of the `+` operator are called **operands**
- you can add an `int` to a `float` with no problem: `1.0 + 2`, the result will be a float `3.0`
- anytime a `float` is added to a number, the result is another `float`
- adding two integers together always results in an `int`

subtraction:

- subtraction is performed with the `-`operator: `3 - 2`, or `5.0 - 3`
- whenever one of the operands is a `float`, the result is also a `float`
- subtracting two integers always results in an `int`

- `-` operator is also used to denote negative numbers: `-3`
- you can subtract a negative number from another number: `1 - -3`
- you can surround negative numbers with parentheses to make it clearer: `1 - (-3)` will result in `4`

multiplication:

- to multiply two numbers, use the `*` operator: `3 * 3`, `2 * 8.0`
- type of number you get from multiplication follows the same rules as addition and subtraction

division:

- the `/` operator is used to divide two numbers: `9 / 3` will result in `3.0`, `5.0 / 2` will result in `2.5`
- division with the `/` operator always returns a `float`
- to make sure that you get an integer after dividing two numbers, you can use `int()` to convert the result: `int(9 / 3)`

  note: `int()` discards any fractional part of the number, so `int(5.0 / 2)` will result in `2`
- division by 0 will result in ZeroDivisionError: try for yourself!!!

integer division:

- **integer division** operator (`//`), also known as the **floor division** operator: `9 // 3` results in `3`, `5.0 // 2` results in `2.0`, `-3 // 2` results in `-2`
- the `//` operator first divides the number on the left by the number on the right and then rounds **down** to an integer

exponents:

- you can raise a number to a power using the `**` operator: `2 ** 2` results in `4`, `2 ** 3` results in `8`
- exponents can be ints, floats, or negative numbers

modulus operator:

- the `%` operator, or the **modulus**, returns the remainder of dividing the left operand by the right operand: `5 % 3` results in `2`, `20 % 7` results in `6`
- `1 % 0` gives ZeroDivisionError: think why!!!

arithmetic expressions:

- you can combine operators to form complex expressions
- an **expression** is a combination of numbers, operators, and parentheses that Python can compute, or **evaluate**, to return a value
- e.g. `-1 + (-3 * 2 + 4)`
- rules for evaluating expressions are the same as in everyday arithmetic: `*`, `/`, `//`, and `%` operators all have equal **precedence** (priority)

math functions:

- `round()` — for rounding numbers to some number of decimal places, or round a number to the nearest integer

  `round(2.3)`

  output: 2

  `round(2.7)`

  output: 3

  note: `round()` has unexpected behavior when the number ends in `.5`

  to round a number to a given number of decimal places pass a second argument to `round()`:

  `round(3.14159, 3)`

  output: 3.142

  note second number should only be an integer

- `abs()` — finds absolute value

  The absolute value of a number *n* is just *n* if *n* is positive and *-n* if *n* is negative. For example, the absolute value of `3` is `3`, while the absolute value of `-5` is `5`.

- `pow()` — raises to a power

to raise a number to a power you can use either the `**` operator or `pow()` function

`pow()` takes two arguments. The first argument is the **base**, or the number to be raised to a power, and the second argument is the **exponent**, or the power to which the number is to be raised

```
pow(2, 3)
```

output: 8

table with operators

Operators	**Syntax**	**Functioning**
+	x + y	Addition
–	x – y	Subtraction
*	x * y	Multiplication
/	x / y	Division
//	x // y	Quotient
%	x % y	Remainder
**	x ** y	Exponentiation

precedence:

Operators	Meaning
`()`	Parentheses
`**`	Exponent
`*`, `/`, `//`, `%`	Multiplication, Division, Floor division, Modulus
`+`, `-`	Addition, Subtraction

what if operators have same precedence?

answer: Associativity is the order in which an expression is evaluated that has multiple operators of the same precedence. These operators have left-to-right associativity. E.g. if you have this expression: `print(5 * 2 // 3)` multiplication and floor division have the same precedence. Hence, the left one is evaluated first.
output: 3

You can assign same value to several variables, here’s how:

```
# Initialize x, y, z to 1
x = y = z = 1
```

note: in this case assignment operators do not have associativity

more on strings:

We can get individual characters in a string using square brackets [].

```
course = ‘Python’
course[0] # returns the first character
course[1] # returns the second character
course[-1] # returns the first character from the end
course[-2] # returns the second character from the end
```

try for yourself!!! note: print() is not included, so nothing will be printed, what you need to do is to include print() for lines 2-5

We can slice a string using a similar notation:

```
course[1:5]
```

The above expression returns all the characters starting from the index position of 1 to 5 (but excluding 5). The result will be **ytho**

exercises

Save each python program for which you wrote any code in this format: p3_exerciseNum.py e.g. p3_20.py

1. Is this number an integer or a float?
    a. 15

b. 3.14159
c. 25_000
d. 1_234.567
e. 10e3
f. 0.5
g. 100_000_000
h. 2.5e-2
i. 7
j. 9.87654321

2. Now using type() function see the results of a previous exercise. For each number print out its data type.

3. What is the result of the operation and what is the data type of the result
   a. 4 + 7
   b. 15 - 6
   c. 5 + 3.2
   d. 10.5 - 2
   e. 3 * 5
   f. 10 / 3
   g. 10 // 3
   h. 2 ** 4
   i. 17 % 5

4. Now check your results in a program, create variables and assign operations from previous exercise, then print the results and using type() function determine the data type of each result.

5. Create a variable called `first_number` and assign it the user's input. Prompt the user to enter an integer by displaying a message like "Enter an integer: ". Convert the user's input from a string to an integer using the `int()` function. Create another variable called `second_number` and assign it any integer value of your choice. Create a third variable called `sum` and calculate the sum of `first_number` and `second_number`. Print `first_number` and `second_number`, and the value of `sum` using the `print()` function. Then, print the data type of `sum` using the `type()` function and the `print()` function.

6. Create a variable called `first_number` and assign it the user's input. Prompt the user to enter a floating-point number by displaying the message "Enter a floating-point number: ". Convert the user's input from a string to a float using the `float()` function. Create another variable called `second_number` and assign it any float value of your choice. Create a third variable called `difference` and calculate the difference between `first_number` and `second_number`. Print `first_number`, `second_number`, and the value of `difference` using the `print()` function. Then, print the data type of `difference` using the `type()` function and the `print()` function.

7. Arithmetic Operations on User Input
    a. Create two variables, `num1` and `num2`.
    b. Prompt the user to enter a value for `num1` by displaying the message "Enter the first number: ".
    c. Prompt the user to enter a value for `num2` by displaying the message "Enter the second number: ".
    d. Convert the user input to the appropriate data type (integer or float) using the `int()` or `float()` function.
    e. Perform the following arithmetic operations:
        - Calculate the sum of `num1` and `num2` and assign it to a variable called `sum_result`.
        - Calculate the subtraction of `num1` and `num2` and assign it to a variable called `sub_result`.
        - Calculate the multiplication of `num1` and `num2` and assign it to a variable called `mul_result`.

- Calculate the division of `num1` and `num2` and assign it to a variable called `div_result`.

f. Print out the results of each operation using the `print()` function.

8. Ask the user to enter a number. Calculate and print the square and cube of that number.

9. You have been saving money in your piggy bank. You saved $10 on Monday, $15 on Tuesday, and $20 on Wednesday. Create appropriate variables, use only one arithmetic expression in this exercise to calculate the total amount of money you saved over the three days. Print the result with a message.

10. You went shopping for school supplies and bought a notebook for $5, pencils for $1.50 each, and a ruler for $2. Create appropriate variables, use only one arithmetic expression in this exercise to calculate the total cost of your school supplies. Print the result with a message.

11. You are running a lemonade stand and sold 10 cups of lemonade for $2.50 each. Calculate the total amount of money you earned by multiplying the number of cups sold by the price per cup. Print the result.

12. Given the radius of a circle as 5 centimeters, calculate the circumference and the area of the circle. Print both results. Use the formulas: circumference = 2 * π * radius and area = π * radius^2. (You can use an approximate value of π as 3.14)

13. You want to calculate your monthly expenses based on your weekly budget. Ask the user for the weekly budget and the number of weeks in a month. Convert the user input to the appropriate data type and then calculate the total monthly expenses by multiplying the weekly budget by the number of weeks. Print the total monthly expenses.

14. You are going shopping and there is a 20% discount on all items. Ask the user for the original price of an item. Convert the user input to the appropriate data type and then calculate the discounted price by subtracting 20% of the original price. Print the discounted price with a message.

15. You have a temperature in Celsius and you want to convert it to Fahrenheit. Ask the user to enter a temperature in Celsius and convert it to Fahrenheit using the formula:

    Fahrenheit = (Celsius * 9/5) + 32.

    Print the converted temperature in Fahrenheit.

16. You have a duration in minutes and you want to convert it to hours and minutes. Ask the user to enter a duration in minutes and convert it to hours and minutes using integer division and the modulus operator. Print the duration in hours and minutes.

17. You are looking to buy a house and want to estimate the price based on the size of the house. Ask the user to enter the square footage of the house and the price per square foot in a specific area. Calculate and print the estimated price of the house.

18. You and your friends are renting an apartment and want to split the monthly rent evenly. Ask the user to enter the total monthly rent and the number of friends. Calculate and print each person's share of the rent. Don’t forget to include yourself in calculation.

19. You went to a restaurant and want to calculate the total bill including the tip. Ask the user for the total bill amount and the percentage of the tip. Convert the user input to the appropriate data types and calculate the tip amount by multiplying the bill amount by the tip percentage divided by 100. Print the total bill amount including the tip.

20. You are traveling abroad and want to convert your local currency to the currency of the country you're visiting. Follow these steps to write a program:
    a. Ask the user to enter an amount of their local currency. Display a message like "Enter the amount of your local currency: ".
    b. Ask the user to enter the exchange rate to the foreign currency. Display a message like "Enter the exchange rate to the foreign currency: ".
    c. Convert the user input to the appropriate data type using the `float()` function.
    d. Calculate the value of the equivalent amount in the foreign currency by multiplying the amount by the exchange rate.
    e. Print out the converted amount in the foreign currency.

21. Imagine you are going on a shopping spree and you have a certain budget. Create a variable called `budget` and assign it a value representing the amount of money you have. Then, create variables for the prices of three items you want to buy: `item1_price`, `item2_price`, and `item3_price`. Calculate the total cost of the items by adding the prices together and assign it to a variable called `total_cost`. Finally, print the total cost of items and how much money you have left. You can choose to either receive input from the user, or specify data in a program, for now don't worry if you go into minus with your budget.

22. You have a recipe that serves four people, but you need to scale it up to serve a larger group. Follow these steps:
    a. Create a variable called `recipe_servings` and assign it the original number of servings (4).
    b. Ask the user to enter the desired number of servings by displaying a message like "Enter the number of servings you need: ".
    c. Convert the user input to an integer using the `int()` function and assign it to a variable called `desired_servings`.
    d. Calculate the scaling factor by dividing `desired_servings` by `recipe_servings`.
    e. Create variables for each ingredient's quantity in the original recipe.

f. Multiply each ingredient quantity by the scaling factor to get the adjusted quantities.

g. Print out the adjusted quantities for each ingredient.

23. You want to track your fitness progress by calculating your body mass index (BMI). Ask the user to enter their weight in kilograms and their height in meters. Convert the user input to the appropriate data type using the `float()` function. Calculate the BMI by dividing the weight by the square of the height (BMI = weight / (height * height)) and assign it to a variable called `bmi`. Print the calculated BMI, and find on the internet what each BMI means, print it out as well as a part of your output.

24. You are planning a road trip and want to calculate the total travel time based on the distance and average speed. Ask the user to enter the distance to be traveled in kilometers and the average speed in kilometers per hour. Convert the user input to the appropriate data type using the `float()` function. Calculate the travel time by dividing the distance by the average speed and assign it to a variable called `travel_time`. Print the travel time in hours and minutes by converting it to the appropriate format. Try to think how to convert a float number to hours and minutes firstly on your own. If you need some help, here is the note: to convert travel_time to hours all you need to do is to convert a float into an int using int() function. To know the rest of the minutes you can use this formula: int((travel_time - hours) * 60).

25. You are organizing a pizza party and need to order pizzas to satisfy the attendees. Follow these steps:

    a. Ask the user to enter the number of attendees. Display a message like "Enter the number of attendees: ".

    b. Convert the user input to an integer using the `int()` function.

    c. Determine the number of pizzas needed by dividing the number of attendees by the average slices per pizza.

    d. Since you cannot order a fraction of a pizza, round up the result to the nearest whole number.

- To round up without using the `math` module, you can use the `round()` function and add 0.5 to the result before rounding.
- For example: `num_pizzas = round(result + 0.5)`

e. Print the number of pizzas to be ordered.

By using the `round()` function and adding 0.5 to the result before rounding, you can achieve a similar effect as rounding up with the `math.ceil()` function.

26. Given the string `word = "Hello"`, use the concept of accessing individual characters in a string to print the following:
    - The first character of the string.
    - The last character of the string.
    - The third character of the string.
    - The second-to-last character of the string.

27. Given an email address as a string (e.g., "example123@gmail.com"), use the concept of string slicing to print the username part of the email address. The username is the part before the "@" symbol, you can use slicing not only by specifying concrete index, but also using variables. You need to print part before “@” so what you can do is to find this symbol using `index()` function. Create a variable and assign it to this expression: email.index("@") where email is the name of your variable where you store email address. Now you store the index of symbol “@” and you can use string slicing by specifying the range from the beginning of the string (index 0) up to (but not including) the "@" symbol index.

28. Given the string `sentence = "I love programming in Python"`, use the concept of string slicing to print the following:
    - The word "love"
    - The word "programming"
    - The word "Python"

29. Given the encrypted message `message = "saedfdsvnewnpotavutrpe"`, use the concept of accessing individual characters to extract the hidden message by accessing the 2nd, 4th, 8th, 10th, 12th, 15th, 18th, 20th, and 22nd characters in the string. Remember that the indices in a string begin with 0, so you need to subtract 1 from the given characters. Print the decrypted message.

EXTRA TASK: Picnic Planner

You are planning a picnic and need to create a program to calculate the details and costs. Write a program that prompts the user to enter the following information:

a. Number of people: Ask the user to enter the total number of people attending the picnic.
b. Food cost: Ask the user to enter the total cost of food for the picnic.
c. Drink cost: Ask the user to enter the total cost of drinks for the picnic.
d. Location: Ask the user to enter the location of the picnic.

Using the provided information, calculate and display the following information:

a. Total cost: Calculate the total cost of the picnic by adding the food cost and drink cost.
b. Cost per person: Calculate the cost per person by dividing the total cost by the number of people, and round it to two decimal places using the `round()` function.
c. Message: Display a message thanking the attendees for joining the picnic and providing the location and cost per person.

Convert user input to correct data types if needed, use variables correctly and write appropriate messages to interact with user!

Here's just an example:

```
Welcome to the Picnic Planner!

Please enter the number of people attending the picnic: 20
Please enter the total cost of food for the picnic: 150
Please enter the total cost of drinks for the picnic: 80
Please enter the location of the picnic: Central Park

Based on the information provided:
- The total cost of the picnic is $230.00
```

```
- The cost per person is $11.50

Thank you for joining us at the picnic in Central Park!
```