Week 10: COMP-801 - Integrated Computing Practice

Agenda

- Methods: instance, class, and static
- Parameters: positional, keyworded, optional
- 📖 git commands, git branches, and git workflow
- Preparing for H3 and team project

Instance Methods

- Define instance objects' behavior
- Bound to the instance object and called on insatnce objects (using dot notation)
- Depend on **instance attributes** of the class
  - Defined and initialized in the class constructor
  - Accessed and modified by **instance methods**

Class Methods

- Define the behavior of the class, independent of the behavior that pertains to instance objects
- Bound to the class and called on the class itself (using dot notation)
- Depend on **class attributes**
    - Defined and initialized OUTSIDE any of the method definitions
    - Accessed and modified by **class methods**, but also other types of methods

Static Methods

- Similar to a module-level functions
- Encapsulated in a class definition because they offer services related to the class purpose
- Not bound to an instance object or the class itself
- Do not use or need instance or class attributes
- Can be called on either an instance object or the class itself

Instance and Class Methods

- Which attribute is **class attribute**?
- Which attribute is **instance attribute**?

```
class Weight:
  bucket_weight = 10
  def __init__(self, weight):
    self.weight = weight

  def __str__(self):
    return f"weight={self.weight}"

  @classmethod
  def change_bucket_weight(cls, new_b_w):
    cls.bucket_weight = new_b_w
    return cls.bucket_weight

w_obj = Weight(100)
print(w_obj)
print(Weight.change_bucket_weight(15))
```

Static Methods

- What's the purpose of a static method?

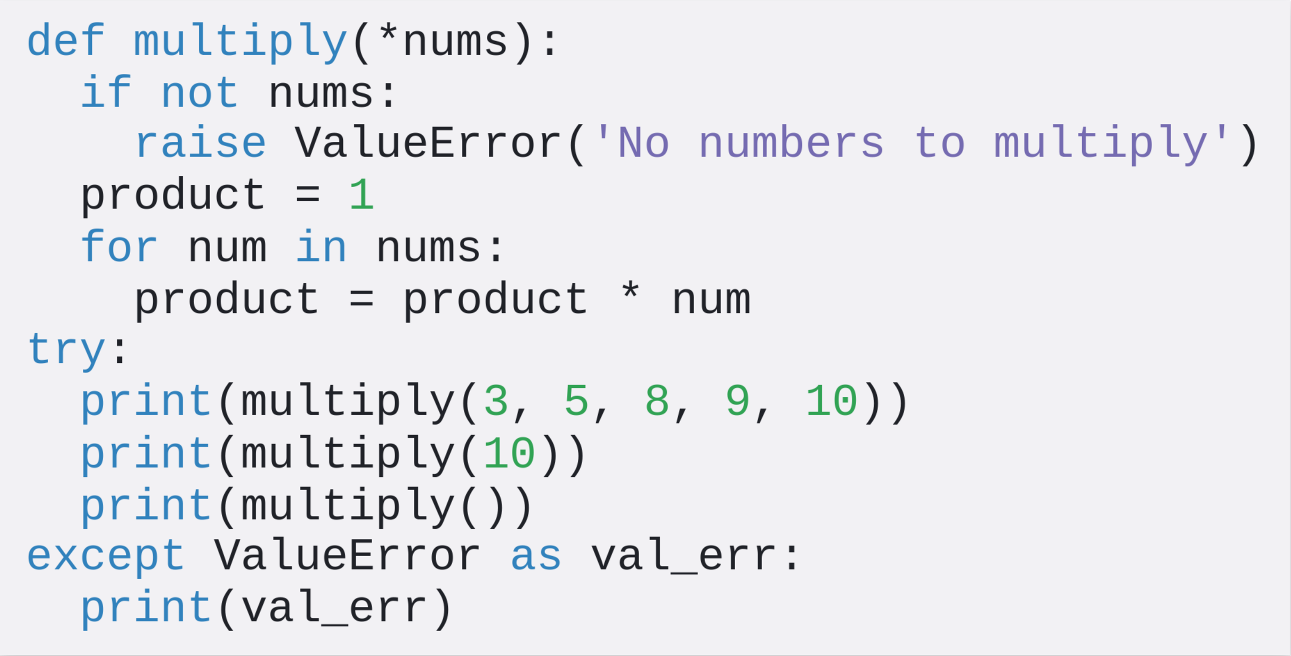
```
class Weight:

 def __init__(self, weight):
    self.weight = weight

  def __str__(self):
    return f"weight={self.weight}"

  @staticmethod
  def convert_lb_to_kg(weight_lb):
    weight_kg = weight_lb * 0.453592
    return weight_kg
```

Positional, Keyword, and Optional Parameters

```python
def divide_or_halve(divident, dividor=2):
  return float(dividend) / dividor
```

```python
# arguments passed to positional parameters
divide_or_halve(6, 3)
# default 2 used for optional parameter
divide_or_halve(6)
# arguments passed to keyword parameters
divide_or_halve(dividend=6, dividor=3)
# keyword arguments can be in any order
divide_or_halve(dividor=3, divident=6)
```

Arbitrary Positional Parameters



```
def multiply(*nums):
  if not nums:
    raise ValueError('No numbers to multiply')
  product = 1
  for num in nums:
    product = product * num
try:
  print(multiply(3, 5, 8, 9, 10))
  print(multiply(10))
  print(multiply())
except ValueError as val_err:
  print(val_err)
```

Arbitrary Keyword Parameters

```
def fruit_qty(some_fruit, **basket):
  if not basket:
      raise ValueError('No basket')
  if some_fruit not in basket:
      raise KeyError(f'No {some_fruit} in {basket}')
  return basket[some_fruit]
```

```
print(fruit_qty('oranges', nuts=2, oranges=3))
```

Errors with Arbitrary Keyword Parameters

```python
# client code
try:
  fruit_qty('apples')
except ValueError as val_err:
  print(val_err)
```

```python
try:
  fruit_qty('apples', nuts=2, oranges=3)
except KeyError as key_err:
  print(key_err)
```

Is Python Compiled or Interpreted?

- What's a compiler?
- What's an interpreter?
- What are .py files?
- What's in __**pycache**__ directory?
- What are .pyc files?
- What is the Python shell?

Is Python Compiled or Interpreted?

Is Python a compiled language or interpreted language?

📖 Version Control With `git`

1. How do we create a local repository?
2. What is a commit?
3. How do we prepare to commit changes made in the working directory of the repository?
4. What are the type of changes that you can commit?
5. What should the commit message say?
6. How do we see the commit history?

📖 git Remotes

7. How do we see remote connections for a repository?
8. How do we create a remote connection?
9. How do we synchronize our local repository with the remote?

Preparing for H3 and Team Project

- Examine StackOverflow 2023 survey data
  - https://survey.stackoverflow.co/2024/
- Extract small subsets of the survey: 5, 10, 50 entries
  - To be able to create tests cases
- Learn how to process and analyzed large data sets
- New Python library module: `csv`