UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

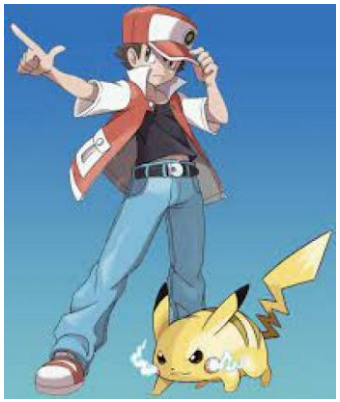
Python Programming for Data Science and Engineering

Ph.D. School in Information Engineering
A.Y. 2024/2025

Stefano Tortora
stefano.tortora@unipd.it
Intelligent Autonomous Systems Laboratory (IAS-Lab)
Dept. of Information Engineering (DEI)
University of Padua, Italy

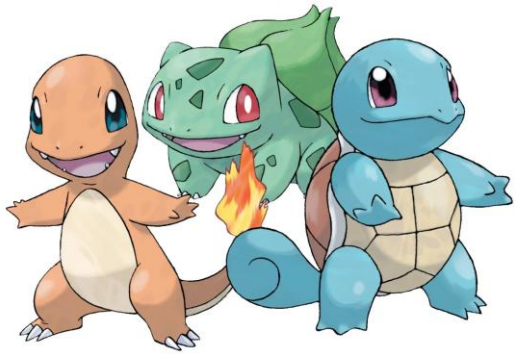
Create a Pokemon simulator

Pokémon is a series of video games developed by Game Freak and published by Nintendo and The Pokémon Company under the *Pokémon* media franchise



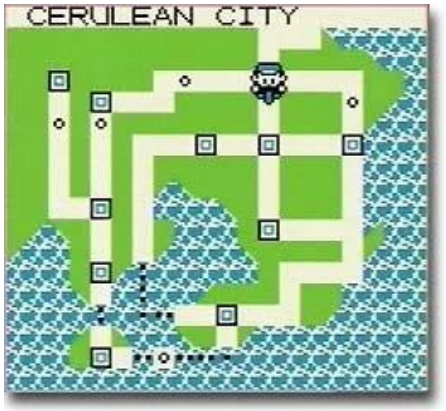
Pokemon Trainer

Pokemon characters



Items

Settings



Assignment 1

The aim of **Assignment 1** is to build the main elements composing the *Pokemon Game*. Thus, to complete the assignment you must:

1. Create the main classes (Pokemon Trainer, Pokemon Character)
2. Manage the combat (implement the useMove() method)
3. Implement a Main() to:
   - ❑ Ask the user to create his Pokemon Trainer;
   - ❑ Instantiate some Pokemon Characters;
   - ❑ Collaudate the combat function.

Explanations on how to perform these steps are provided in the following slides.

Create the main classes

Pokemon Trainer Class

The pokemon trainer represents your main character in the fictional world of the game controlled by the player.

Attributes

Name
Pokemon list (max 6)
Items (initially empty)

Pokemon Character Class

The pokemon characters are the main agents involved in the pokemon fights.
They can be:

Player's pokemon: pokemon characters controlled by the player and included in the Pokemon list.

Opponent pokemon: pokemon characters encountered during the exploration and controlled by the game.

Attributes

Name
Level (set to 1)
Types
baseStats
moves (max 4, no repetitions)
Current Health Points (HP)

N.B. Of course, you can create additional classes and/or add other attributes, if you want!

Manage baseStats and Moves

Every pokemon character has the following *baseStats*:

HP → maximum HPs
ATTACK → value used to deal physical attacks
DEFENSE → value used to defend from physical attacks
SPEED → determines the pokemon velocity
SPECIAL → value used for attacking and defending with special moves

```
"national_pokedex_number": 1,
"name": "bulbasaur",
"types": ["grass", "poison"],
"baseStats": {"hp": 45, "attack":
49, "defense": 49, "speed": 45,
"special": 65}
```

In addition, every pokemon character can learn a maximum of 4 different pokemon **moves**. Each move has the following characteristics:

NAME
TYPE
CATEGORY (physical or special)
POWER
ACCURACY (probability of hitting the target)
PP (max number of times a move can be used)

```
#PHYSICAL ATTACK
{"name": "tackle", "type": "normal",
"category": "physical", "power": 35,
"accuracy": 0.95, "pp": 35}
#SPECIAL ATTACK
{"name": "water gun", "type":
"water", "category": "special",
"power": 40, "accuracy": 1.0,
"pp": 25}
```

Manage the Combat

For the Pokemon class, create the method useMove(): given the selected move and the opponent pokemon, the method should first check if a probability value is lower than the "accuracy" value of the move. If the check is True, compute the damage dealt by the pokemon to the opponent, according to the following formula, and apply that damage to the opponent.

$$damage = floor\left(\left(\frac{2 * level + 10}{250} * \frac{atk}{def} * pow + 2\right) * modifier\right)$$

$$modifier = stability * effect * critical * luck$$

- **atk** = is "attack" value of attacking pokemon if move is physical, otherwise is "special"
- **def** = is "defense" value of opponent pokemon if move is physical, otherwise is "special"
- **pow** = "power" value of the move
- **stability** = is 1.5 if pokemon "types" contain the move "type", otherwise is 1
- **effect** = always equal to 1
- **critical** = is 2 if (probability < "speed" / 512), otherwise is 1
- **luck** = random value in [0.85; 1)

Manage probability

If you want to test if something is going to happen with a certain probability, you can start with creating a random float number in the range from 0.0 (included) to 1.0 (not included) and check if the number is lower a certain probability.
In the following, you have a simple example.

```
import random

probability = 0.8
success = random.random() < probability
if success:
    print('You succeeded in doing something. Great job!')
else:
    print('Not succeeded. What a pity, try again!')
```

By looking to the documentation of the random library you can find other useful functions for computing random values (e.g., randint(), uniform()).

Create player

In the main section of your program, create a player character (i.e., Pokemon trainer) by asking the user:

- The name of the character
- To select one among three starter pokemons (Bulbasaur, Charmender, Squirtle) to add to the trainer's Pokemon list

```
options = ['Bulbasaur', 'Charmender', 'Squirtle']

print('Options:')
for i, opt in enumerate(options):
    print(i, ':', opt)
choice = int(input('Choose option:'))
```

Then, collaudate the useMove() method of the selected Pokemon against an enemy Pokemon.
When a Pokemon uses a move (independently if it is successful or not), reduce the PPs of that move. If the PP of the move is 0, the Pokemon cannot use that move.

Starter pokemons

Bulbasaur

"national_pokedex_number": 1,
"name": "bulbasaur",
"types": ["grass", "poison"],
"baseStats": {"hp": 45, "attack": 49, "defense": 49, "speed": 45, "special": 65}
"moves" : ["tackle", "razor leaf"]

Charmender

"national_pokedex_number": 4,
"name": "charmander",
"types": ["fire"],
"baseStats": {"hp": 39, "attack": 52, "defense": 43, "speed": 65, "special": 50}
"moves" : ["tackle", "ember"]

Squirtle

"national_pokedex_number": 7,
"name": "squirtle",
"types": ["water"],
"baseStats": {"hp": 44, "attack": 48, "defense": 65, "speed": 43, "special": 50}
"moves" : ["tackle", "water gun"]

Basic moves

tackle

```
"name": "tackle", "type": "normal", "category": "physical",
"power": 35, "accuracy": 0.95, "pp": 35
```

razor leaf

```
"name": "razor leaf", "type": "grass", "category": "physical",
"power": 55, "accuracy": 0.95, "pp": 25
```

ember

```
"name": "ember", "type": "fire", "category": "special",
"power": 40, "accuracy": 1.0, "pp": 25
```

water gun

```
"name": "water gun", "type": "water", "category": "special",
"power": 40, "accuracy": 1.0, "pp": 25
```