

COM S 327, Spring 2022

Programming Project 1.07

Loading Pokémon

We've parsed in a number of data files that specify certain details about how to create pokémon. Now we're actually going to load them into our game. We'll add the ability to encounter them, but we'll save battling and capturing for next week.

We're going to simplify many of the mechanics of the pokémon main series games (MSGs) in our implementation. The full mechanics are just too tedious, which would result in a lot of code and not much learning. If correct (or more correct) mechanics are important to you, you're welcome to implement something more detailed than presented here.

When walking through tall grass, each move the PC makes has a certain chance of encountering a pokémon. In the MSGs, different areas of the world will have different kinds of pokémon in the grass, as well as other types of terrain (desert, caves, water) that may spawn pokémon, as well. We'll simplify this by allowing any (literally any!) pokémon to spawn in any tall grass, this includes fully evolved forms and legendary and mythical pokémon. We've parsed in details on 1092 pokémon; we'll select from them on a uniform distribution, which means that you have an equal chance of encountering a pidgey and a mewtwo. As for the probability of encountering a pokémon in the tall grass, something like 10% seems reasonable.

The levels of encountered pokémon will increase with the manhattan distance from the origin. When distance is less than or equal to 200, minimum level is 1 and maximum level is $\text{distance} / 2$ (integer division). When distance exceeds 200, minimum level becomes $(\text{distance} - 200) / 2$ and maximum level is 100.

Encountered pokémon will know up to two moves. The known moves are restricted to the pokémon's level-up learnset and will only be less than two if the pokémon has fewer than two moves in its level-up learnset at its level.

The pokémon in our database are grouped in many ways. The `version_group_id` in `pokemon_moves.csv` is a unique identifier of a pokémon game, and (I think) a value of 19 is the only one for which all pokémon have learnsets. We'll use that one, so that we can use all of the pokémon. Again, you're welcome to do something different here, but it will increase the complication of your game somewhat.

To find the level-up moveset of a pokémon, you'll want `pokemon_moves.version_group_id` equal to 19, `pokemon_moves.pokemon_id` equal to `pokemon.species_id` and `pokemon_moves.pokemon_move_method_id` equal to 1. From these, select your two moves from a uniform distribution.

Pokémon stats (attack, defense, etc.) are given in `pokemon_stats.csv`, which we didn't parse last week, so you'll need to parse that in now. The mapping of stat numbers to names is given in `stats.csv`. Each pokémon has an additive deviation from its species' base stats, known as IVs (individual values). IVs for each stat are uniformly distributed in the range [0,15]. Generate IVs for each spawned pokémon for each of HP, attack, defense, speed, special attack, and special defense.

The mechanics of pokémon gender are somewhat involved (designed to reduce storage, see https://bulbapedia.bulbagarden.net/wiki/Individual_values). Some pokémon don't even have genders. We'll simplify this by making all species have equal probability of being male or female. As with everything else here, you're welcome to implement the more complex mechanic if you like.

Pokémon have a $1/8192$ chance of being shiny, based on their IVs (see the above link). Again, simplify this by ignoring the IVs and simply using, e.g., `rand() % 8192 == 0`. Feel free to reduce that modulus to increase your shiny rate.

In order to level up your pokémon, we'll apply the following formulae:

$$HP = \lfloor \frac{((base + IV) \times 2) \times level}{100} \rfloor + level + 10$$

$$Otherstat = \lfloor \frac{((base + IV) \times 2) \times level}{100} \rfloor + 5$$

where *base* is the pokémon species' base stat, *IV* is its IV for the given stat and *level* is the pokémon's level. Again, we're simplifying an MSG mechanic here by ignoring EVs (effort values), which normally add an additional boost to pokémon growth. See

<https://bulbapedia.bulbagarden.net/wiki/Stat> for the deets.

When encountering pokémon, you'll add a placeholder function for the battle and capture sequence in which you'll print the pokémon's level, stats, and moves and pause for input (escape, any key, etc.).

A final note: There are probably only two of you (maybe three) who know less about pokémon than me. Please let me know if I got something qualitatively wrong here, or if you think of a better simplification to MSG mechanics than I've proposed.