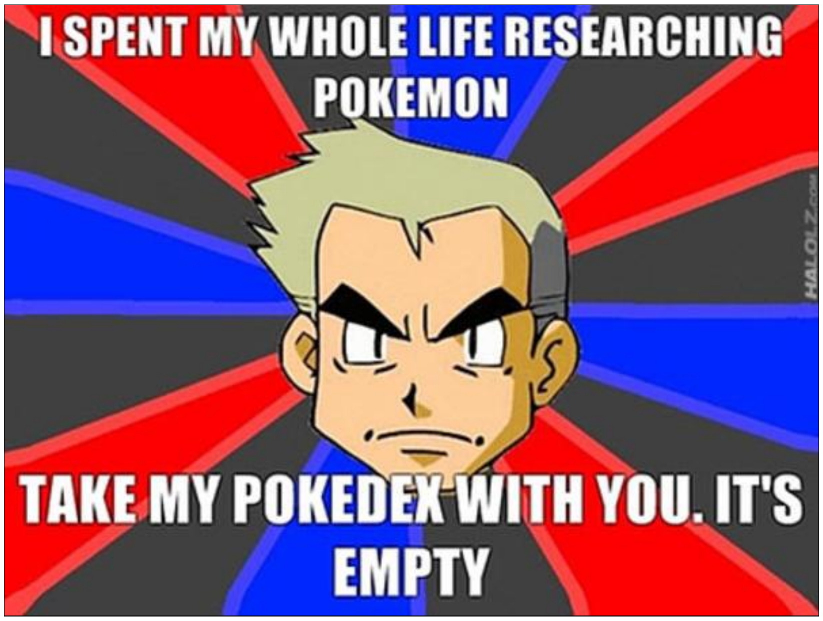
COP 3502

Project 4

Due November 15, 2016


I SPENT MY WHOLE LIFE RESEARCHING POKEMON
TAKE MY POKEDEX WITH YOU. IT'S EMPTY
HALOLZ.COM

Background

Word has spread about the great work you've been doing creating software to predict monster captures. A certain Professor Oak from Pallet Town is particularly impressed with your work, and is interested in enlisting your aid in an exciting new endeavour. He heard that programs written in Java can run on millions of different devices, so he wants you to develop a portable, text-based Pokedex program that his field agents can use in their varying computer systems that are used in many different regions.

Objective

To be the very best, like no one ever was.

Your Assignment

Armed with your newly gained knowledge of Arrays and Objects, your task is to develop a program which makes use of simple but well-structured Pokemon and Pokedex classes in order for field agents to keep track of the strange new Pokemon they encounter on their journies.

Your Pokedex will have the advanced functionality that Professor Oak's field agents have come to expect: They will be able to:

- add new species of Pokemon to their Pokedex
- list of all the Pokemon that have been stored
- list an individual Pokemon's statistics
- sort their Pokedex
- The program that you create and submit **must** contain three files, named exactly as follows: `Pokemon.java`, `Pokedex.java`, and `Project4.java`, which will hold your `main()` method. Each of the .java files and their associated classes are described below:

The Pokemon Class

Your Pokemon class, which is stored in Pokemon.java, defines what constitutes an object of type Pokemon. The class **must** have the following members:

- Fields for the following attributes:
  - `species`
  - `attack`
  - `defense`
  - `speed`
- Modifier methods (**used in testing, not in menu**):
  - `public void grow(int boost)`
    - This method increases `attack` based on the given `boost`
  - `public void harden(int boost)`
    - This method increases `defense` based on given `boost`
  - `public void haste(int boost)`
    - This method increases `speed` based on the given `boost`
- "Getter"/"Setter" methods for each attribute:
  - `public String getSpecies()`
  - `public void setSpecies(String spc)`
      - Names should consist of **one-word** strings ONLY.
  - `public int getAttack()`
  - `public void setAttack(int atk)`
  - `public int getDefense()`
  - `public void setDefense(int def)`
  - `public int getSpeed()`
  - `public void setSpeed(int spd)`
- An appropriate constructor:
  - `public Pokemon(String species)`
  - The constructor must initialize all 4 fields of the Pokemon object with the constructor's parameters:
    - `species = species Passed in`

- `attack = (Length of species name * 4) + 2`
- `defense = (Length of species name * 2) + 7`
- `speed = (Length of species name * 3) +  5`

The Pokedex Class

The Pokedex class, which is stored in Pokedex.java, will define what constitutes an object of type Pokedex. The Pokedex class **must** have the following members:

- An array of Pokemon objects, which will hold any pokemon that the user adds to it.

  - You may have any additional fields that may aid the rest of the class's functionality (strongly recommended). It is recommended to keep a variable in the Pokedex that allows you to keep track of the number of Pokemon you have stored so far.
  - The user will have to input the size of the array

- Member methods (More details on the usage each method are provided later in the project description- You **must** have the same method definitions as the ones provided below):

  - `public String[] listPokemon()`

    - This method returns an array of Strings containing the names of every Pokemon stored in the Pokedex. The Pokedex class has a field that is an array of Pokemon objects. This method will look at each Pokemon stored in the Pokedex object's Pokemon array, access the name of the species, and save it into an Array of Strings. The `String[]` output should be a list containing the names of all Pokemon species currently saved in the Pokedex, with no empty indexes. For example, if your Pokedex holds 20 Pokemon and you have only stored 10, your `String[]` returned should have a length of 10, not 20. If the Pokedex is currently empty, return null. See example runs for details.

  - `public boolean addPokemon(String species)`

    - This method takes a String parameter called species and stores it in the Pokedex. It does this by adding it to the **next free index** in the Pokedex's Pokemon[] array. Return `true` if the operation was successful, otherwise return `false`.

      - You should check the Pokedex's Pokemon[] array to see if it already contains the species of Pokemon which you are trying to add (**regardless of case differences**). If it does, do **NOT** add the Pokemon to the Pokedex - instead, print the word "**`Duplicate`**" and return `false`

- You should also design this method so that it does not add the Pokemon if the Pokedex is already at the maximum size. If the Pokedex object's Pokemon[] array is already full, it should instead print the word "**`Max`**" and return `false`

- See the sample run for more details.

- `public int[] checkStats(String species)`

  - This method returns an array of 3 integers containing the attack, defense, and speed of the pokemon specified by a `species` string argument, **in that order.**
  - If the given Pokemon is not in the dex, return `null.` When checking whether the given Pokemon exists, use **case-insensitive string comparisons.**
  - See sample run for more details.

- `public void sortPokemon()`

  - This method implements a simple **Selection Sort**. It will sort the elements in the Pokedex's Pokemon array alphabetically based on the Pokemon's species names.
  - See https://en.wikipedia.org/wiki/Selection_sort for details.
  - Hint: Use the method `compareToIgnoreCase()`

- An appropriate constructor:

  - A simple constructor with a single integer parameter that defines the size of its Pokemon array

    -
      ```
      public Pokedex(int size){

           //initialize Pokemon[] array
      }
      ```

The Project4 Class

The Project4 class, stored in Project4.java, will hold the main method of your program. **We will provide `Project4.java` to you.** It is your job not to change it. Project4.java works as follows:

- The main method will prompt the user to input the number of Pokemon they wish their Pokedex to store. Then main() will create a Pokedex object of that size.
- It will then prompt the user for input:

    - 1. List Pokemon
    - 2. Add Pokemon
    - 3. Check a Pokemon's Stats
    - 4. Sort Pokemon
    - 5. Exit

- If the input given is anything other than an integer between 1 and 5, the program will ask for valid input again. See sample run for more details.
- Depending on user input, the program will run the appropriate method from the Pokedex class.

Listing Pokemon

If the user chooses to List Pokemon, the program will call the `listPokemon()` method.

Adding Pokemon

If the user chooses to Add Pokemon to their Pokedex, the program will prompt the user for the name of the new species of Pokemon to be added. Then, it will call `addPokemon()` which will call the constructor to create a new Pokemon object with that name, along with its stats, computed as described above.

Important: The user should NOT be able to add another Pokemon if the Pokedex is already at maximum capacity (the one you specify at the beginning of the program) or if a Pokemon species with the given name (**regardless of case differences**) is already stored in the Pokedex. This should handled by the Pokedex class's `addPokemon()` method as specified above and shown in the sample run.

Checking a Pokemon's Stats

If the user chooses to Check Pokemon Stats, the program will prompt them for the name of the type of Pokemon they are interested in. Then, the Pokedex object will call its `checkStats()` method to look for Pokemon with that name stored in the Pokedex. As specified earlier and shown in the sample run, If the method does not find a matching Pokemon, it will state so. Again, see the sample run for specifics.

Sorting Pokemon

If the user chooses to Sort Pokemon, the program will call the Pokedex's `sortPokemon()` method. This method will sort the Pokemon in the Pokedex alphabetically by their species name, using a simple **Selection Sort (Recommended)**. The method will also rearrange the Pokemon stored in the Pokedex object's array.

- See https://en.wikipedia.org/wiki/Selection_sort for details on Selection Sort. For more help understanding and implementing this simple sort, YouTube has many great videos explaining it, and you have many TAs that would be happy to help you.

- You may attempt to implement better, more efficient sorting algorithms if you are familiar with them, though you do this at your own risk.

Other Important Notes:
Please make sure to mimic the output in the Sample Runs, you know the drill by now. Also please be sure to name your files appropriately. In addition, make sure to comment your code. If your program needs to be regraded manually, it would benefit your grade if your TA were to actually be able to understand what you are trying to accomplish.

Make sure to read past the sample runs for submission instructions
Sample Runs (input in red):
```
Welcome to your new PokeDex!
How many Pokemon are in your region?: 10
Your new Pokedex can hold 10 Pokemon. Let's start using it!
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 1
Empty
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 2
Please enter the Pokemon's Species: Snorlax
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 1
1. Snorlax
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 2
Please enter the Pokemon's Species: Artucino
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
```

```
4. Sort Pokemon
5. Exit
What would you like to do? 2
Please enter the Pokemon's Species: Artucino
Duplicate
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 1
1. Snorlax
2. Artucino
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 2
Please enter the Pokemon's Species: Snorlax
Duplicate
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 4
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 1
1. Artucino
2. Snorlax
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 3
Please enter the Pokemon of interest: Snorlax
The stats for Snorlax are:
Attack: 30
Defense: 21
Speed: 26
1. List Pokemon
```

```
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 3
Please enter the Pokemon of interest: Metagross
Missing
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 1
1. Artucino
2. Snorlax
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 2
Please enter the Pokemon's Species: metagross
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 2
Please enter the Pokemon's Species: Metagross
Duplicate
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 5
```

Run 2:

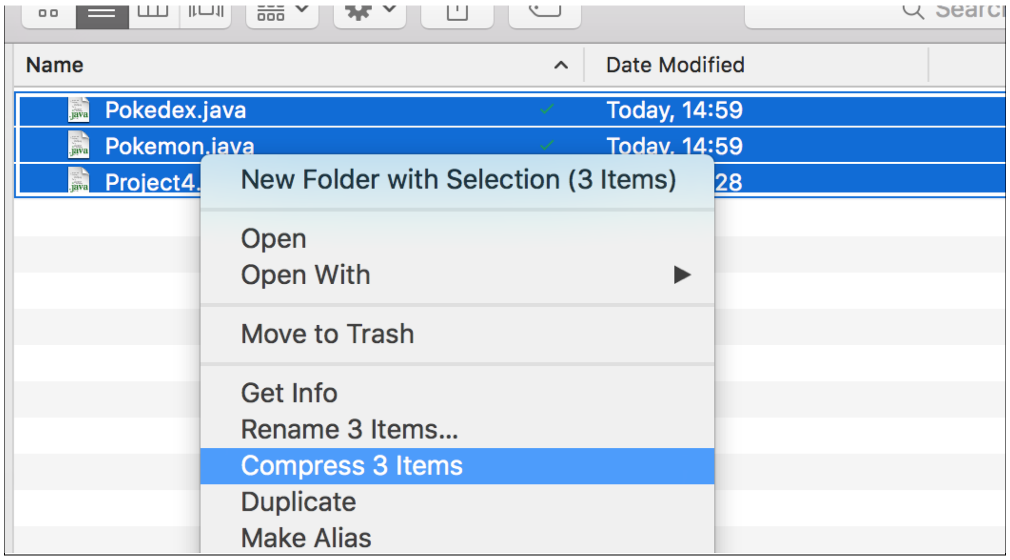
```
Welcome to your new PokeDex!
How many Pokemon are in your region?: 4
Your new Pokedex can hold 4 Pokemon. Let's start using it!
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 2
Please enter the Pokemon's Species: zapdos
```

```
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 2
Please enter the Pokemon's Species: umbreon
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 2
Please enter the Pokemon's Species: entei
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 2
Please enter the Pokemon's Species: Scyther
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 2
Please enter the Pokemon's Species: scizor
Max
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 1
1. zapdos
2. umbreon
3. entei
4. Scyther
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 4
1. List Pokemon
2. Add Pokemon
```

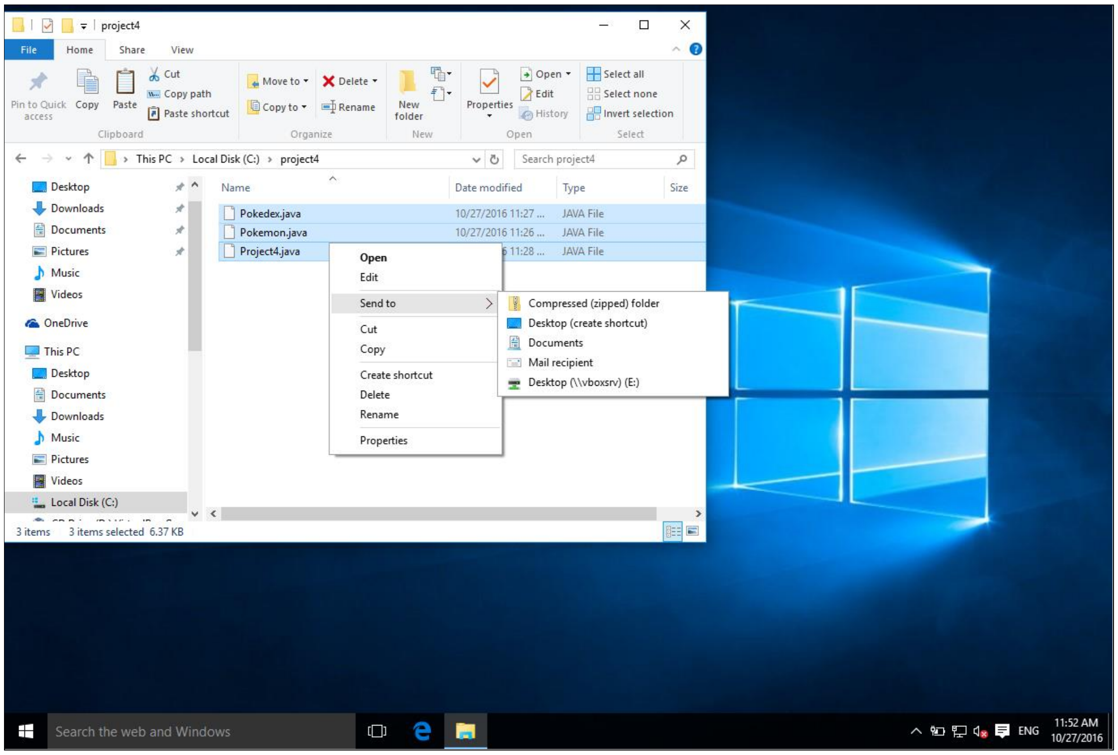
```
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 1
1. entei
2. Scyther
3. umbreon
4. zapdos
1. List Pokemon
2. Add Pokemon
3. Check a Pokemon's Stats
4. Sort Pokemon
5. Exit
What would you like to do? 5
```

Submission Requirements

- Name the classes described above as `Pokemon.java`, `Pokedex.java`, and `Project4.java`
- Please zip the 3 files and send them. Please do NOT create a folder.
- Mac: Select all source files -> right click -> Compress 3 items



- Windows: Select all source files -> right click -> Compressed (zipped) Folder



- Please **rename** the zipped file such that it is named `project4_ulfid.zip` where uflID is the alphanumeric portion of your ufl email account that comes before the **@ufl.edu** part.
- Submit on time using Canvas

If you submit the files without zipping, your project will not be graded.
If your zip file is not named `project4_ulfid.zip`, your project will not be graded.
If you do not name the items above exactly as specified, where specified, your project will not be graded.
It is highly recommended that you test your program piece by piece before assembling your final code. You will have a much easier time if you build your program piece by piece rather than trying to write the entire program. Your methods **should not differ in any way** from the methods given. Using additional prompts, words, abbreviations, etc may result in the grading program taking unnecessary points off and the world's smallest violin being played for your sorrows.
Please be sure to use the sample output to check your work

Grading

- 100% of your program will be graded on implementing all of the mandatory Class and methods follow the instruction. You are expected to generate proper output (from the methods).