BLG252 ASSIGNMENT 3 REPORT

MEHMET ALİ BALIKÇI

150200059

MARCH 2023

Contents

1- INTRODUCTION

This report analyzes a code for two pokemons to fight each other as a game, which includes classes and objects for, Pokemon, Electric Pokemon, Psychic Pokemon, and Arena. It demonstrates how the messages sent to these objects enable the game to be played effectively. For further details, please refer to the code provided in the assignment.

2- METHOD

To analyze the code, the report will utilize the Debug feature of Visual Studio Code and repeatedly use the step info feature to ensure that no level of abstraction is overlooked.

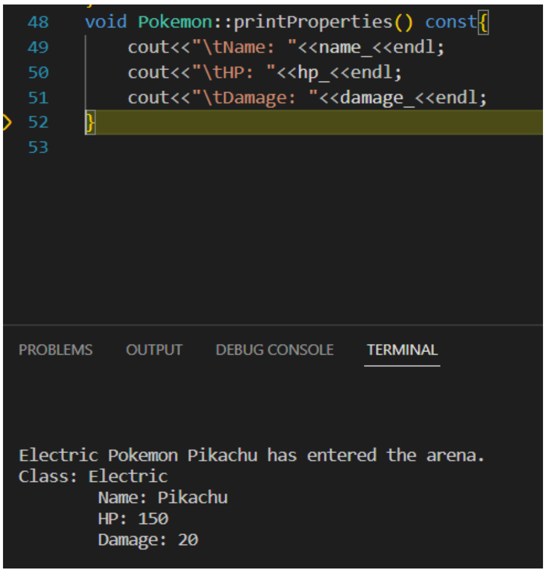
3- IMPLEMENTATION

As a first, the methods and attributes of five classes existing on the code respectively are:

Pokemon Class:

- Attributes:
  - name_ : It keeps the name of pokemon.
  - hp_ : It keeps the level of health of pokemon.
  - damage_: It holds how much a pokemon damage.
  - possibility_of_powerUP_ : It keeps what possibility a pokemon can have powerUP.
  - availability_of_powerUP_ : It holds whether the time of recharge of a pokemon run out.
  - state_of_powerUP_ : It keeps whether a pokemon have a powerUP or not at that moment.
  - is_first_ : It holds whether a pokemon is first to fight or not.
  - recharge_ : It keeps the time of recharge of a pokemon.
- Methods:
  - Pokemon() : It is a constructor. This gets information of name, hp, recharge time, damage, powerUP chance and creates a pokemon.
  - getPokemonName() : It returns the name of pokemon.
  - getHP() : It returns the level of health of pokemon.
  - setHP() : It sets the level of health of a pokemon according to damage the pokemon takes.

    - getDamage() : It returns the amount of damage a pokemon cause.
    - getPossibilityOfPowerUP() : It returns the possibility of powerUP a pokemon can have.
    - getCanPowerUP() : It returns whether a pokemon can have a powerUP or not because of its recharge time.
    - setCanPowerUP() : It sets that a pokemon can have a powerUP since its reharge time runs out.
    - getStateOfPowerUP() : It returns whether a pokemon has a powerUP at that moment or not.
    - setStateOfPowerUP() : It sets whether or not a pokemon has a powerUP at that moment.
    - getIsFirst() : It returns whether a pokemon will fight first or not.
    - setIsFirst() : It sets whether a pokemon will fight first or not.
    - getFainted() : It returns whether a pokemon has fainted or not.
    - printProperties() : It prints the features of a pokemon such as name, the amount of damage, the level of health. Like:



```
void Pokemon::printProperties() const{
    cout<<"\tName: "<<name_<<endl;
    cout<<"\tHP: "<<hp_<<endl;
    cout<<"\tDamage: "<<damage_<<endl;
}
```

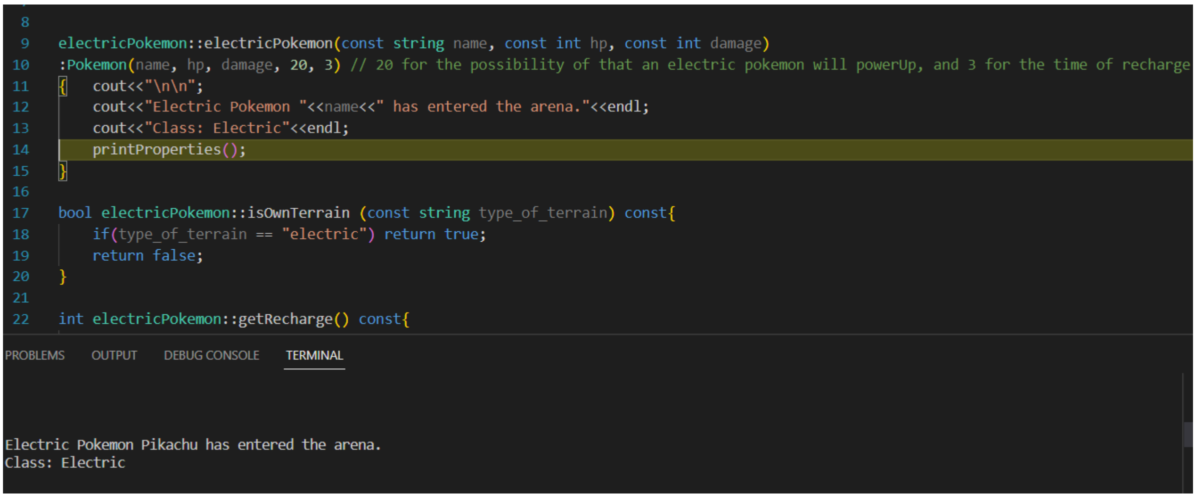
```
Electric Pokemon Pikachu has entered the arena.
Class: Electric
        Name: Pikachu
        HP: 150
        Damage: 20
```

electricPokemon Class:

There are not a lot of attributes or methods in this class as it is inherited from Pokemon class.

- Attributes:
    - There are no attributes in this class.
- Methods:

- electricPokemon() : It is a constructor. It takes the data of name, hp and damage. Then, it creates an electric pokemon. Like:



```cpp
electricPokemon::electricPokemon(const string name, const int hp, const int damage)
:Pokemon(name, hp, damage, 20, 3) // 20 for the possibility of that an electric pokemon will powerUp, and 3 for the time of recharge
{   cout<<"\n\n";
    cout<<"Electric Pokemon "<<name<<" has entered the arena."<<endl;
    cout<<"Class: Electric"<<endl;
    printProperties();
}

bool electricPokemon::isOwnTerrain (const string type_of_terrain) const{
    if(type_of_terrain == "electric") return true;
    return false;
}

int electricPokemon::getRecharge() const{
```

```
Electric Pokemon Pikachu has entered the arena.
Class: Electric
```

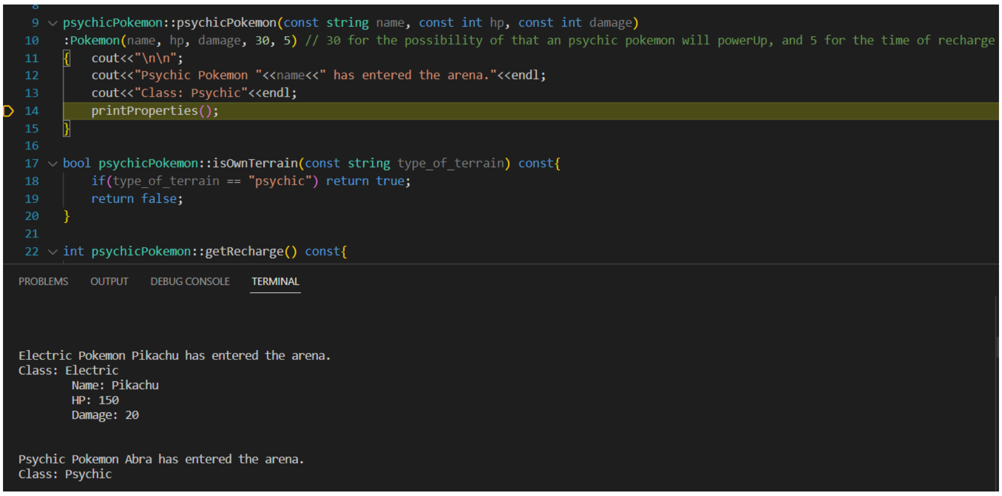
- isOwnTerrain() : It provides the pokemon to know whether it is in its own terrain or not.
- getRecharge() : It returns the time of recharge of an electric pokemon.

psychicPokemon Class:

There are not a lot of attributes or methods in this class as it is inherited from Pokemon class.

- Attributes:
  - There are no attributes in this class.

- Methods:
  - psychicPokemon() : It is a constructor. It takes the data of name, hp and damage. Then, it creates a psychic pokemon. Like:



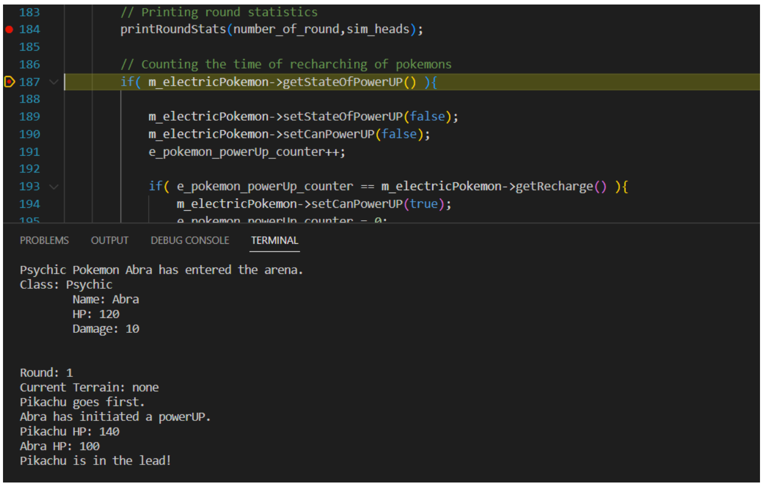
```cpp
psychicPokemon::psychicPokemon(const string name, const int hp, const int damage)
:Pokemon(name, hp, damage, 30, 5) // 30 for the possibility of that an psychic pokemon will powerUp, and 5 for the time of recharge
{   cout<<"\n\n";
    cout<<"Psychic Pokemon "<<name<<" has entered the arena."<<endl;
    cout<<"Class: Psychic"<<endl;
    printProperties();
}

bool psychicPokemon::isOwnTerrain(const string type_of_terrain) const{
    if(type_of_terrain == "psychic") return true;
    return false;
}

int psychicPokemon::getRecharge() const{
```

```
Electric Pokemon Pikachu has entered the arena.
Class: Electric
        Name: Pikachu
        HP: 150
        Damage: 20


Psychic Pokemon Abra has entered the arena.
Class: Psychic
```

- isOwnTerrain() : It provides the pokemon to know whether it is in its own terrain or not.
- getRecharge() : It returns the time of recharge of a psychic pokemon.

Arena Class:

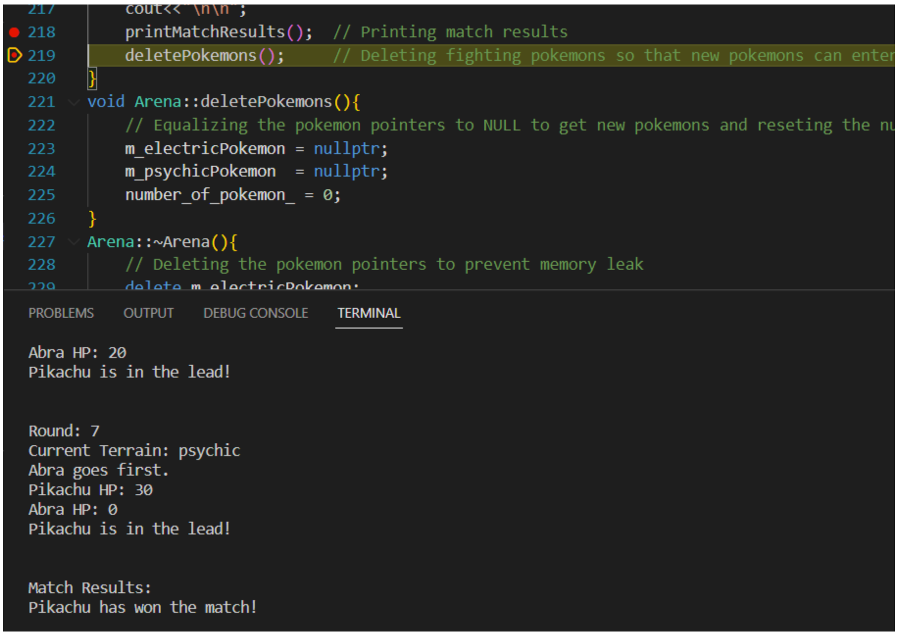
- Attributes:
  - m_electricPokemon : It is a pointer to electricPokemon. It enables a composition relationship between the arena class and the electric pokemon class.
  - m_psychicPokemon : It is a pointer to psychicPokemon. It enables a composition relationship between the arena class and the electric pokemon class.
  - type_of_terrain_ : I keeps which type pokemons figth.
  - number_of_pokemon_ : It holds how many pokemons exist in the arena so that more than two pokemons can not enter the arena.
  - e_pokemon_can_damage_ : It keeps whether an electric Pokemon can damage because it can be confused.
- Methods:
  - addPokemon() : It takes the datas of type, name, hp and damage of pokemons from main, and it creates an electric pokemon and a psychic pokemon according to this information.
  - spawnTerrain() : It spawns terrains like psychic terrain with a 20% possibility, electric terrain with a 20% possibility, and none 60% possibility.
  - restartTerrain() : It restarts the terrain setting it to none whenever pokemons enter the arena.
  - printRoundStats() : It prints the information about what things happened in the round. Like:



```
    // Printing round statistics
    printRoundStats(number_of_round,sim_heads);

    // Counting the time of recharching of pokemons
    if( m_electricPokemon->getStateOfPowerUP() ){

        m_electricPokemon->setStateOfPowerUP(false);
        m_electricPokemon->setCanPowerUP(false);
        e_pokemon_powerUp_counter++;

        if( e_pokemon_powerUp_counter == m_electricPokemon->getRecharge() ){
            m_electricPokemon->setCanPowerUP(true);
            e_pokemon_powerUp_counter = 0;
```

```
Psychic Pokemon Abra has entered the arena.
Class: Psychic
        Name: Abra
        HP: 120
        Damage: 10


Round: 1
Current Terrain: none
Pikachu goes first.
Abra has initiated a powerUP.
Pikachu HP: 140
Abra HP: 100
Pikachu is in the lead!
```

- printMatchResults() : It prints the results of match. Like:

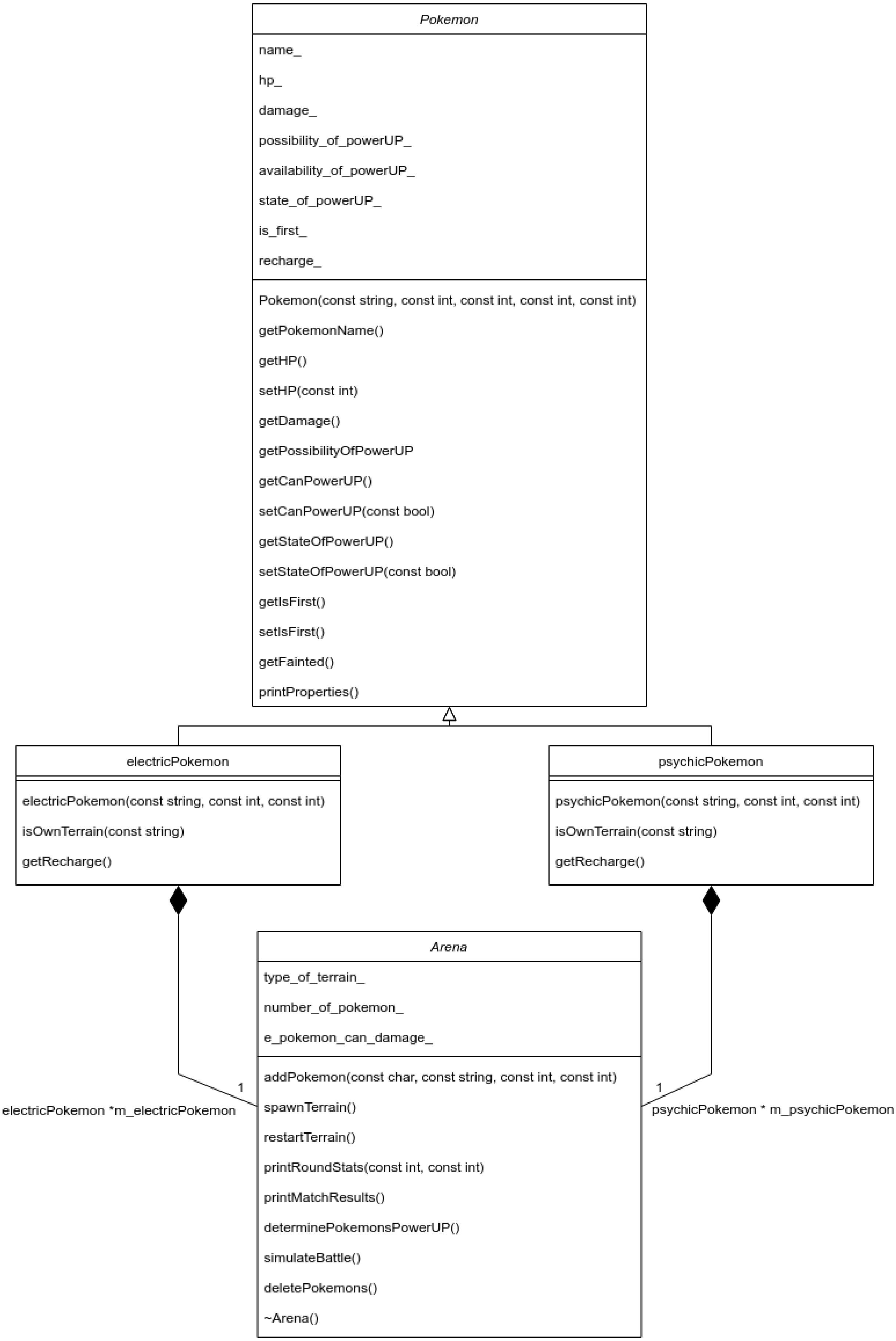


```
    cout<<"\n\n";
    printMatchResults();  // Printing match results
    deletePokemons();     // Deleting fighting pokemons so that new pokemons can enter
}
void Arena::deletePokemons(){
    // Equalizing the pokemon pointers to NULL to get new pokemons and reseting the nu
    m_electricPokemon = nullptr;
    m_psychicPokemon  = nullptr;
    number_of_pokemon_ = 0;
}
Arena::~Arena(){
    // Deleting the pokemon pointers to prevent memory leak
    delete m_electricPokemon;
```

```
Abra HP: 20
Pikachu is in the lead!


Round: 7
Current Terrain: psychic
Abra goes first.
Pikachu HP: 30
Abra HP: 0
Pikachu is in the lead!


Match Results:
Pikachu has won the match!
```

- determinePokemonsPowerUP() : It determines whether pokemons will start a powerUP or not accourding to their powerUP possibilities.
- simulateBattle() : It simulates battle.
- deletePokemons() : It deletes out-matched pokemons so that new pokemons enter the arena.
- ~Arena() : It is a destructor. When the lifespan of an arena class runs out, it deletes m_electricPokemon and m_psychicPokemon pointers so as not to happen a memory leak.

4- UML DIAGRAM


Pokemon
name_
hp_
damage_
possibility_of_powerUP_
availability_of_powerUP_
state_of_powerUP_
is_first_
recharge_
Pokemon(const string, const int, const int, const int, const int)
getPokemonName()
getHP()
setHP(const int)
getDamage()
getPossibilityOfPowerUP
getCanPowerUP()
setCanPowerUP(const bool)
getStateOfPowerUP()
setStateOfPowerUP(const bool)
getIsFirst()
setIsFirst()
getFainted()
printProperties()
electricPokemon
electricPokemon(const string, const int, const int)
isOwnTerrain(const string)
getRecharge()
psychicPokemon
psychicPokemon(const string, const int, const int)
isOwnTerrain(const string)
getRecharge()
Arena
type_of_terrain_
number_of_pokemon_
e_pokemon_can_damage_
addPokemon(const char, const string, const int, const int)
spawnTerrain()
restartTerrain()
printRoundStats(const int, const int)
printMatchResults()
determinePokemonsPowerUP()
simulateBattle()
deletePokemons()
~Arena()
1
electricPokemon *m_electricPokemon
1
psychicPokemon * m_psychicPokemon