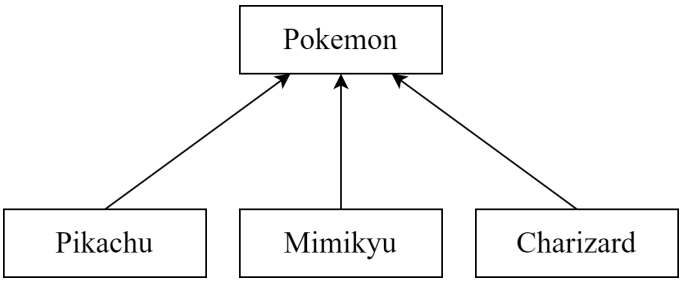
Q6.

For this problem, we will implement inheritance and polymorphism.
The architecture is shown as the following figure.
We have a base class Pokemon, and three derived classes



Class Pokemon

/* Public constructor.
- hp: Pokémon's hp.
- attack: Pokémon's attack.
- defense: Pokémon's attack
- specialAttack: Pokémon's attack
- specialDefense: Pokémon's specialDefense
- speed: Pokémon's speed

*/

- Pokemon(int hp, int attack, int defense, int specialAttack, int specialDefense, int speed)

// Print out all the information about this Pokémon. You can see the format in the sample output.

- void info()

// Pure virtual function.

- virtual void description()

// public function to get the Pokémon's nickname.

- string getNickname()

// Public function to set the Pokémon's nickname.

- void setNickname(string name)

Class Pikachu

// Public constructor, set the default nickname to "Dummy"
Pikachu::Pikachu(int hp, int attack, int defense, int specialAttack, int specialDefense, int speed)

// print out the nickname and the Pokémon's name. e.g. [this->nickname] is Pikachu

```
void Pikachu::description()
```

Class Mimikyu

```
// Public constructor, set the default nickname to “Dummy”
Mimikyu::Mimikyu(int hp, int attack, int defense, int specialAttack, int specialDefense, int speed)

// print out the nickname and the Pokémon’s name. e.g. [this->nickname] is Mimikyu
void Mimikyu::description()
```

Class Charizard

```
// Public constructor, set the default nickname to “Dummy”
Charizard::Charizard(int hp, int attack, int defense, int specialAttack, int specialDefense, int speed)

// print out the nickname and the Pokémon’s name. e.g. [this->nickname] is Charizard
void Charizard::description()
```

Class Trainer

```
private:
    string name;    // The name of the Trainer
    Pokemon* pokemon;  // The Pokémon currently being used by the Trainer.

public:
    // The public constructor
    Trainer(string name): name(name), pokemon(nullptr) {}

    // Attempt to switch out the Pokémon that the Trainer is currently using.
    void changePokemon(Pokemon* pokemon);
```

Input Format

The first line is an integer n, which indicates the number of Pokémon in the program.
And then there will be n line, which format is
[Pokémon’s class], [Pokémon’s nickname], [Hp], [Attack], [Defense], [SpecialAttack], [SpecialDefense], [Speed]
You need to read these data and instantiate each Pokémon using their corresponding class.

Output Format

Print the description, and attributes of Pokémon.
Print whether the Trainer can change the Pokémon (class == Charizard).
You should follow the TODOs in the template.

Sample Input

```
3
Pikachu, pikachu, 142, 117, 101, 112, 112, 156
Mimikyu, mimikyu, 162, 156, 145, 112, 172, 162
Charizard, charizard, 185, 149, 143, 177, 150, 167
```

Sample Output

```
pikachu is Pikachu
> hp: 142
> attack: 117
> defense: 101
> specialAttack: 112
> specialDefense: 112
> speed: 156

mimikyu is Mimikyu
> hp: 162
> attack: 156
> defense: 145
> specialAttack: 112
> specialDefense: 172
> speed: 162

charizard is Charizard
> hp: 185
> attack: 149
> defense: 143
> specialAttack: 177
> specialDefense: 150
> speed: 167

Trainer Red can not use Pikachu
Trainer Red can not use Mimikyu
Trainer Red changes Pokemon successfully
```