Viva3Q1

Question

Modify your `Aran.isValid()` method to accept an input string consisting of alphabets and digits. Digits can be placed anywhere.

There are two types of characters: Main Characters (PTMQ) and Special Characters (BDGH).

Rules for Main Characters:

- P can only be followed by T.
- M can only be followed by M.
- Q can only be followed by none.

However, these rules on Main Characters can be broken due to the presence of Special Characters.

Rules for Special Characters:

- B may or may not exist before P.
- D may or may not exist before T.
- G may or may not exist after M.
- H may or may not exist after Q.

As long as the Special Characters follow the defined rules, the sequence of PTMQ could be random. For example:

```
- TP: false
- TBP: true
- BDT: true
- DTP: false
```

The input string should contain at least one main character, otherwise, it should return false."

Sample Input

```
public static void main(String[] args) {
    String[] testCases = new String[] {
            "PTM", "PTMMTP", "BPTMBPTM", "PT", "PTMQ", "PTMQH", "BPTMQH", "PTMQH",
"BPDTMGQH",
            "PTMQH", "PTMQH", "12PTHQGM", "12PTQHMG", "12PTMGQH",
"PTMQBPTMQBPTMQ", "DTBP123QHMG",
            "BTDP", "TP", "TBP", "BDT", "DTP", "ABCD", "D"
    };

    for (String input : testCases)
```

```
            System.out.printf("%-20s : %s\n", input, Aran.isValid(input));
    }
```

Sample Output

```
PTM                 : true
PTMMTP              : false
BPTMBPTM            : true
PT                  : true
PTMQ                : true
PTMQH               : true
BPTMQH              : true
PTMQH               : true
BPDTMGQH            : true
PTMQH               : true
PTMQH               : true
12PTHQGM            : false
12PTQHMG            : false
12PTMGQH            : true
PTMQBPTMQBPTMQ      : true
DTBP123QHMG         : true
BTDP                : false
TP                  : false
TBP                 : true
BDT                 : true
DTP                 : false
ABCD                : false
D                   : false
```

Question 2

In the context of a Pokemon battle, you are tasked with modifying the method `determineWinner(String name)` to ascertain the winners from each type of Pokemon when facing a specific opponent. The method should return an array with a size of 3, representing the names of Pokemon that can triumph over the specified opponent from different types.

The expected format of the array is as follows:

```
String[] winners = {Flame-type Pokemon, Grass-type Pokemon,
Water-type Pokemon};
```

However, there is a condition where no Pokemon from a particular type can defeat the opponent. In such cases, you should return the String "`no Pokemon`" at the specific index in the array.

Test Program

```
public class PokemonTest {
    public static void main(String[] args) {
        Pokemon moltres = new Pokemon("Moltres","Flame",
85.0);
        Pokemon servine = new Pokemon("Servine","Grass",
60.0);
        Pokemon charmander = new Pokemon("Charmander",
"Flame", 92.0);
        Pokemon pansage = new Pokemon("Pansage", "Grass",
55.0);
        Pokemon araquanid = new Pokemon("Araquanid", "Water",
74.0);
        Pokemon flareon = new Pokemon("Flareon", "Flame",
65.0);
        Pokemon squirtle = new Pokemon("Squirtle", "Water",
63.0);
        Pokemon wooper = new Pokemon("Wooper", "Water",
42.0);

        Pokemon[] PokemonList = {moltres, servine,
                charmander, pansage, araquanid, flareon,
squirtle,
                wooper};
        PokemonSortingSystem sortingSystem = new
PokemonSortingSystem(PokemonList);
        System.out.println("List of Pokemon after Sorting:
");
        for (Pokemon pokemon : PokemonList) {
```

```
            System.out.println(pokemon);
        }
        System.out.println();


determineWinners("Squirtle",sortingSystem.determineWinner("Sq
uirtle", PokemonList),PokemonList);

determineWinners("Charmander",sortingSystem.determineWinner("
Charmander", PokemonList),PokemonList);

determineWinners("Flareon",sortingSystem.determineWinner("Fla
reon", PokemonList),PokemonList);
    }

    public static void determineWinners(String opponentName,
String[] winners, Pokemon[] PokemonList){
        System.out.println("Pokemon effective against " +
opponentName + ":");

System.out.printf("%15s%15s%15s\n","Flame","Grass","Water");
        for (String winner : winners) {
            System.out.printf("%15s",winner);
        }
        System.out.println("\n");
    }
}
```

Sample Output

```
List of Pokemon afterSorting:
Wooper
Pansage
Servine
Squirtle
Flareon
Araquanid
Moltres
Charmander

Pokemon effective against Squirtle:
          Flame          Grass          Water
     Charmender        Servine      Araquanid
```

```
Pokemon effective against Charmander:
          Flame          Grass          Water
     no Pokemon     no Pokemon      Araquanid

Pokemon effective against Flareon:
          Flame          Grass          Water
     Charmander     no Pokemon      Araquanid
```