In [1]:
```
import pandas as pd
from itertools import combinations
```

In [2]:
```
def rSubset(arr, r):

    # return list of all subsets of length r
    # to deal with duplicate subsets use
    # set(list(combinations(arr, r)))
    return list(combinations(arr, r))
```

In [3]:
```
Pokemon = pd.read_csv('./Data/gen9_pokemon_stats.csv')
Pokemon = Pokemon[(Pokemon.Pokemon != "Iron Moth") &
                  (Pokemon.Pokemon != "Flutter Mane") & (Pokemon.Pokemon != "Armarouge") & (Pokemon.Pokemon != "Iron Jugulis")&
                  (Pokemon.Pokemon != "Iron Valiant") & (Pokemon.Pokemon != "Iron Bundle") & (Pokemon.Pokemon != "Sandy Shocks") &
                  (Pokemon.Pokemon != "Iron Hands")& (Pokemon.Pokemon != "Slither Wing") & (Pokemon.Pokemon != "Palafin") &
                  (Pokemon.Pokemon != "Koraidon") & (Pokemon.Pokemon != "Iron Thorns") & (Pokemon.Pokemon != "Quaquaval") &
                  (Pokemon.Pokemon != "Brute Bonnet") & (Pokemon.Pokemon != "Great Tusk")&(Pokemon.Pokemon != "Roaring Moon")&
                  (Pokemon.Pokemon != "Iron Treads") & (Pokemon.Pokemon != "Scream Tail")& (Pokemon.Pokemon != "Meowscarada")
                 ]

Pokemon = Pokemon[(Pokemon.Pokemon != "Miraidon")]

# Pokemon = Pokemon[(Pokemon.Pokemon != "Chien-Pao") & (Pokemon.Pokemon != "Wo-Chien")&(Pokemon.Pokemon != "Ting-Lu") &
#                   (Pokemon.Pokemon != "Chi-Yu")
#                  ]


# Pokemon = Pokemon.to_dict()
```

In [4]:
```
#Pokemon.info()
```

In [5]:
```
# Key is defenders weak to
Attacker = {
            'Electric':['ground'],
            'Normal':['fighting'],
            'Ghost':['dark'],
            'Dragon':['ice', 'fairy'],
            'Fairy':['poison', 'steel'],
            'Bug':['flying', 'rock','fire'],
            'Dark':['bug','fighting', 'fairy'],
            'Fighting':['flying', 'psychic', 'fairy'],
            'Fire':['ground', 'rock', 'water'],
            'Flying':['rock', 'electric', 'ice'],
            'Grass':['flying', 'poison', 'bug', 'fire', 'ice'],
            'Ground':['water', 'grass', 'ice'],
            'Ice':['fighting', 'rock', 'steel', 'fire'],
            'Poison': ['ground', 'psychic'],
            'Psychic':['bug', 'ghost', 'dark'],
            'Rock':['fighting', 'ground', 'steel', 'water', 'grass'],
            'Steel':['fighting', 'ground','fire'],
            'Water':['grass', 'electric'],
           }
Attacker
```

Out[5]:
```
{'Electric': ['ground'],
 'Normal': ['fighting'],
 'Ghost': ['dark'],
 'Dragon': ['ice', 'fairy'],
 'Fairy': ['poison', 'steel'],
 'Bug': ['flying', 'rock', 'fire'],
 'Dark': ['bug', 'fighting', 'fairy'],
 'Fighting': ['flying', 'psychic', 'fairy'],
 'Fire': ['ground', 'rock', 'water'],
 'Flying': ['rock', 'electric', 'ice'],
 'Grass': ['flying', 'poison', 'bug', 'fire', 'ice'],
 'Ground': ['water', 'grass', 'ice'],
 'Ice': ['fighting', 'rock', 'steel', 'fire'],
 'Poison': ['ground', 'psychic'],
 'Psychic': ['bug', 'ghost', 'dark'],
 'Rock': ['fighting', 'ground', 'steel', 'water', 'grass'],
 'Steel': ['fighting', 'ground', 'fire'],
 'Water': ['grass', 'electric']}
```

In [6]:
```
LineUp = []
for Type in Attacker:
    L_Type = Type.lower()
    Top = Pokemon[Pokemon['Type 1'] == L_Type].sort_values('Special Attack', ascending=False)['Pokemon'].head(2).to_numpy()
    for names in Top:
        if names not in LineUp:
            LineUp.append(names)

    Top = Pokemon[Pokemon['Type 1'] == L_Type].sort_values('Attack', ascending=False)['Pokemon'].head(1).to_numpy()
    for names in Top:
        if names not in LineUp:
            LineUp.append(names)

    Top = Pokemon[Pokemon['Type 2'] == L_Type].sort_values('Special Attack', ascending=False)['Pokemon'].head(2).to_numpy()
    for names in Top:
        if names not in LineUp:
            LineUp.append(names)

    Top = Pokemon[Pokemon['Type 2'] == L_Type].sort_values('Attack', ascending=False)['Pokemon'].head(1).to_numpy()
    for names in Top:
        if names not in LineUp:
            LineUp.append(names)

len(LineUp)
```

Out[6]:
```
49
```

In [7]:
```
SixLineup = rSubset(LineUp, 6)
len(SixLineup)
```

Out[7]:
```
13983816
```

In [8]:
```
Pokemon_dict = Pokemon.to_dict()
```

In [9]:
```
Check_List=[]
Has_ran = "No"
for Type in Attacker:
```

```
    print(Type)
    Weakness = 'Weakness-' + str(Type)
#     print(Weakness)

    if Check_List != [] or Has_ran != "No":
        SixLineup = Check_List
        Check_List=[]

    Has_ran = "Yes"

    for eachlineup in SixLineup:
#         Check = "No"
        for each in eachlineup:

            num = list(Pokemon_dict['Pokemon'].keys())[list(Pokemon_dict['Pokemon'].values()).index(each)]
            if ((Pokemon_dict['Type 1'][num] in Attacker[Type] or Pokemon_dict['Type 2'][num] in Attacker[Type])) and (Pokemon_dict[Weakness][num] <= 1):
                #
#                 Check = "Yes"
                Check_List.append(eachlineup)

#                 print(Pokemon_dict['Type 1'][num], Pokemon_dict['Type 2'][num],Attacker[Type] )
                break
#         if Check == "Yes":
#             continue

len(Check_List)
```

```
Electric
Normal
Ghost
Dragon
Fairy
Bug
Dark
Fighting
Fire
Flying
Grass
Ground
Ice
Poison
Psychic
Rock
Steel
Water
```

Out[9]: 104568

In [10]:
```
Final_Dict={}

for count in range(len(Check_List)):
    total_sum = 0
    for Type in Attacker:
        power = 0
        Weakness = 'Weakness-' + str(Type)
        for pokemon in Check_List[count]:
            num = list(Pokemon_dict['Pokemon'].keys())[list(Pokemon_dict['Pokemon'].values()).index(pokemon)]
            if ((Pokemon_dict['Type 1'][num] in Attacker[Type] or Pokemon_dict['Type 2'][num] in Attacker[Type])) and Pokemon_dict[Weakness][num] <= 1:
```

```
                if power < max(Pokemon_dict['Special Attack'][num], Pokemon_dict['Attack'][num]):
                    power = max(Pokemon_dict['Special Attack'][num], Pokemon_dict['Attack'][num])
                    price_pokemon = pokemon
#         print(Type, power, price_pokemon)
        total_sum = total_sum + power
    Final_Dict[str(count)]= total_sum

# Final_Dict[0:4]
```

In [18]:
```
Final_Dict_1 = sorted(Final_Dict.items(), key=lambda x:x[1], reverse=True)
Final_Dict_1[0:5]
```

Out[18]:
```
[('86412', 2260),
 ('87652', 2260),
 ('28206', 2250),
 ('91434', 2250),
 ('86425', 2245)]
```

In [24]:
```
ID= 86412
print(Check_List[ID])

for Type in Attacker:
    power = 0
    price_pokemon = None
    Weakness = 'Weakness-' + str(Type)
    for pokemon in Check_List[ID]:
        num = list(Pokemon_dict['Pokemon'].keys())[list(Pokemon_dict['Pokemon'].values()).index(pokemon)]
        if ((Pokemon_dict['Type 1'][num] in Attacker[Type] or Pokemon_dict['Type 2'][num] in Attacker[Type]) and Pokemon_dict[Weakness][num] <= 1) :
            if power < max(Pokemon_dict['Special Attack'][num], Pokemon_dict['Attack'][num]):
                power = max(Pokemon_dict['Special Attack'][num], Pokemon_dict['Attack'][num])
                price_pokemon = pokemon
#                 print(Type, pokemon , Pokemon_dict[Weakness][num], Weakness)
    print(Type, Attacker[Type])
    print("     ",price_pokemon, power)
```

```
('Arboliva', 'Baxcalibur', 'Chien-Pao', 'Kingambit', 'Flamigo', 'Ting-Lu')
Electric ['ground']
      Ting-Lu 110
Normal ['fighting']
      Flamigo 115
Ghost ['dark']
      Kingambit 135
Dragon ['ice', 'fairy']
      Chien-Pao 120
Fairy ['poison', 'steel']
      Kingambit 135
Bug ['flying', 'rock', 'fire']
      Flamigo 115
Dark ['bug', 'fighting', 'fairy']
      Flamigo 115
Fighting ['flying', 'psychic', 'fairy']
      Flamigo 115
Fire ['ground', 'rock', 'water']
      Ting-Lu 110
Flying ['rock', 'electric', 'ice']
      Baxcalibur 145
Grass ['flying', 'poison', 'bug', 'fire', 'ice']
      Baxcalibur 145
Ground ['water', 'grass', 'ice']
```

```
      Baxcalibur 145
Ice ['fighting', 'rock', 'steel', 'fire']
      Kingambit 135
Poison ['ground', 'psychic']
      Ting-Lu 110
Psychic ['bug', 'ghost', 'dark']
      Kingambit 135
Rock ['fighting', 'ground', 'steel', 'water', 'grass']
      Kingambit 135
Steel ['fighting', 'ground', 'fire']
      Flamigo 115
Water ['grass', 'electric']
      Arboliva 125
```