Corrigé du TD de Java n°1 : Attrapez-les tous !

1. Pouvez-vous créer une variable **p2** de type Pokemon, dont le nom sera "Porygon", le type "Epileptique", et la force d'attaque 42 ?

```
Pokemon p2;
p2 = new Pokemon();

p2.nom = "Porygon";
p2.type = "Epileptique";
p2.puissance=42;
```

2. Maintenant, nous devons essayer d'organiser nos Pokemons. On peut commencer par faire un tableau de Pokemons : ce tableau permettra de stocker tous les Pokemons que l'on possède. Écrivez le code permettant de construire un tableau de 100 Pokemons.

```
int i;
Pokemon[] tab;
tab = new Pokemon[100];

for(i=0; i<100; i=i+1)
{
	tab[i] = new Pokemon();
}
```

3. Imaginons que l'on possède un tableau (nommé **collection**) de Pokemons. Ce tableau est sensé lister tous les Pokemons que vous possédez. Écrivez un morceau de code permettant d'afficher le nom de tous les Pokemons de votre tableau (réfléchissez au prototype de la fonction d'abord).

```
AfficherNomPokemon(Pokemon[] collection)
{
	for(i=0; i<100; i=i+1)
	{
		Afficher(collection[i].nom);
	}
}
```

4. Pour effacer un Pokemon du tableau, on changera son nom par "" (la chaîne vide). Écrivez un algorithme qui prend en paramètre un tableau de Pokemon et une position (un entier) et qui efface le Pokemon placé à cette position (réfléchissez au prototype de la fonction d'abord).

```
EffacerPokemon(Pokemon[] collection, int position)
{
	collection[position].nom = "";
}
```

5. On souhaite que le tableau ne possède pas de trou, c'est à dire de case vide en plein milieu. Recommencez la question précédente en prenant cette contrainte en considération.

Pour résoudre ce problème, on va copier le Pokemon en dernière position du tableau à la place de notre Pokemon actuel, et effacer le dernier Pokemon du tableau.

```
EffacerPokemon(Pokemon[] tab, int position)
{
      int position_dernier_pokemon =  tab.length-1;
      tab[position].nom = tab[position_dernier_pokemon].nom;
      tab[position].type = tab[position_dernier_pokemon].type;
      tab[position].puissance = tab[position_dernier_pokemon].puissance;

      //Puis, on efface le dernier Pokemon du tableau
      tab[position_dernier_pokemon].nom = "";
}
```

Le problème ici, c'est que si on appelle une seconde fois la fonction, on se rend compte que le dernier Pokemon du tableau n'est pas celui à la dernière case, mais celui à l'avant dernière case. On doit donc "chercher" dans notre tableau le dernier Pokemon grâce à ce petit morceau de code à greffer dans la fonction précédente :

```
int position_dernier_pokemon =  tab.length-1;

Tant que (tab[position_dernier_pokemon].nom == "")
{
       position_dernier_pokemon = position_dernier_pokemon – 1;
}
```

6. De plus, le tableau est trié, et l'on souhaite conserver l'ordre établi entre les Pokemons. Recommencez la question 4 en prenant cette contrainte supplémentaire en considération.

Alors là, il faut décaler chaque case du tableau d'un cran et effacer le dernier Pokemon du tableau.

```
EffacerPokemon(Pokemon[] tab, int position)
{
      int i;

      for(i=position; i<=tab.length-2; i=i+1)
      {
            tab[position].nom = tab[position+1].nom;
            tab[position].type = tab[position+1].type;
            tab[position].puissance = tab[position+1].puissance;
      }

      tab[tab.length-1].nom="";
}
```

7. Un maître Pokemon perd des Pokemons, mais il lui arrive d'en gagner. Comment feriez-vous pour ajouter un Pokemon à votre collection ?

Deux cas de figure : le tableau peut contenir de nouveaux Pokemons, ou bien le tableau est plein. Dans le premier cas, on rajoute le Pokemon à la fin. Sinon, on doit construire un nouveau tableau avec une case en plus et tout recopier.

```
AjouterPokemon(Pokemon[] tab, Pokemon p)
{
      int position_dernier_pokemon =  tab.length-1;

      while (tab[position_dernier_pokemon].nom == "")
      {
```

```
            position_dernier_pokemon = position_dernier_pokemon - 1;
      }

      if (position_dernier_pokemon != tab.length - 1)
      {
            tab[position_dernier_pokemon].nom = p.nom;
            tab[position_dernier_pokemon].type = p.type;
            tab[position_dernier_pokemon].puissance = p.puissance;
      }
      else
      {
            int i;
            Pokemon tab2[];
            tab2 = new Pokemon[tab.length+1];
            for(i=0; i<tab.length; i=i+1)
            {
                  tab2[i]=tab[i];
            }

            tab2[tab2.length-1] = p;  //et que faire de tab2 après ?
                                      //on pourrait faire tab=tab2...
      }
}
```

8. Que pensez-vous des tableaux pour gérer une collection d'éléments qui peut changer de taille (ajout et suppression d'éléments) ? Voyez-vous une solution plus élégante pour gérer ce problème ?

Ce n'est pas pratique... Les listes seront plus adaptées.

9. Utilisez la solution plus élégante pour répondre aux questions 4 à 7.