
 Institut Sabadell	UF4. Programació Orientada a Objectes	 Curs: 2020/2021
	<i>Moképon - Part 1</i>	

Presentació

Benvingut a Mokepon. En aquest exercici crearàs un joc basat amb capturar i lluitar amb Mokepon, per a aprendre a programar en orientació a objectes

POO (Programació Orientada a Objectes)

La POO és un paradigma de programació consistent en pensar com si fos la vida real. En comptes de tenir un main que fa accions en nom d'objectes, cada objecte és representat en una instància d'una classe.

Podem pensar que una classe és un contenidor de variables (que li direm atributs) i funcions (que li direm mètodes)

Fem una nova classe Mokepon

```
public class Mokepon {

    //atributs del pokemon

    String nom;

    int nivell;

    int atk;

    int def;

    int vel;

    public void diguesNom() {



        //per a accedir a un atribut fem this.nomDeLAtribut

        //this es una paraula especial que fa referencia a nosaltres mateixos.

        System.out.println(this.nom);

    }

}
```

 Institut Sabadell	UF4. Programació Orientada a Objectes	 Curs: 2020/2021
	<i>Moképon - Part 1</i>	

i ara fem una altra classe amb un main

```
public class Test {

    public static void main(String[] args) {

        //podem crear un nou Mokepon perquè existeix la classe i ara es un nou tipus
        //de variable

        //el new Mokepon() es el seu constructor

        Mokepon mikachu = new Mokepon();

        //podem accedir als seus atributs interns amb el punt

        mikachu.nom = "Mikachu";



        //també podem accedir als seus mètodes interns amb el punt

        mikachu.diguesNom();

    }
}
```

Assegura't que funciona bé. Un cop ho has fet, amplia el Mokepon fent el següent

1. Afegeix un nou atribut enter "exp", i un altre enter "hp_max", i un "hp_actual"
2. Fes un nou mètode a Mokepon atorgarExperiencia(int exp_atorgada) que sumi la experiència atorgada a la exp actual. Assegura't que és pública.
3. Si exp passa de 100, fes que es restin 100 d'experiència i la funció cridi a un nou mètode Mokepon.pujarNivell() a on augmenta el nivell en 1, hp_max un valor entre 0 i 5, i l'atac, defensa i velocitat un valor entre 0 i 2 amb un random.
4. Prova aquests mètodes en el main

 Institut Sabadell	UF4. Programació Orientada a Objectes	 Curs: 2020/2021
	<i>Moképon - Part 1</i>	

Constructors

T'hauràs adonat que quan volem crear una nova variable d'una classe (a partir d'ara, un objecte), sempre fem el new. Ho hauràs vist amb el new Scanner(System.in) o amb el new Random(). A vegades es posa algo a dintre dels parèntesis i a vegades no. En realitat quan fem això estem cridant al constructor de la classe.

El constructor és una funció que s'executa al crear un objecte d'aquesta classe. Una classe pot tenir infinits constructors, sempre que els paràmetres (que com ja saps, és allò que li passem a una funció amb els parèntesis) que li arribin siguin diferents.

Totes les classes tenen per defecte el constructor estàndard new NomDeClasse() que et crea un objecte buit. Anem a modificar-lo i crear-ne un altre

```
//fixa't que el constuctor no té retorn, ja que el que retorna es la propia
classe

public Mokepon() {

    this.nom = "Sense definir";

    this.nivell = 1;

    this.atk = 1;

    this.def = 1;

    this.vel = 1;

}

//podem crear múltiples constructors sempre que tinguin paràmetres diferents

public Mokepon(String nom) {

    //this.nom fa referència a la variable global i nom a la local que hem passat
per paràmetre

    this.nom = nom;
}
```

```
this.nivell = 1;

this.atk = 1;

this.def = 1;

this.vel = 1;

}

public Mokepon(String nom, int nivell) {

    //aquest constructor crida al constructor de nom primer, per això el
    this(nom), amb el que farà tot el que el constructor de nom digui

    this(nom);
    //podem fer fors dintre d'un constructor com si fos una funció normal

    for (int i = 1; i < nivell; i++) {

        //podem cridar a funcions de la nostra propia classe amb el this



        this.pujarNivell();

    }

}
```

i ho provem al main

```
Mokepon missingNo = new Mokepon();
System.out.println(missingNo.nom);
Mokepon marmander = new Mokepon("Marmander");
System.out.println(marmander.nom+" "+marmander.atk);
Mokepon mulmasaur = new Mokepon("Mulmasaur", 5);
System.out.println(mulmasaur.nom+" "+mulmasaur.atk);
```

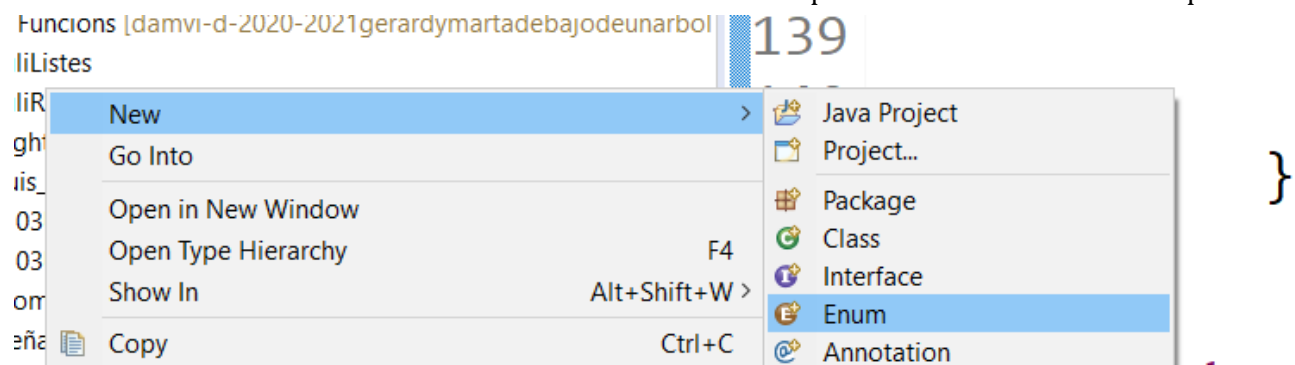
 Institut Sabadell	UF4. Programació Orientada a Objectes	 Curs: 2020/2021
	<i>Moképon - Part 1</i>	

Amplia els constructors fent el següent

1. Modifica els constructors per a afegir també el camp hp_max que has afegit. El valor per defecte d'un nivell 1 serà 10
2. Afegeix un nou constructor a on li passis nom, nivell, hp_max, atk, def, vel, i et crei un pokemon amb aquests valors

Enums

Anem a afegir Tipus als Mokepons. Els Mokepons tan sols poden tenir tres tipus, Foc, Aigua i Planta. Podríem representar els tipus amb una String, però podria ser que algú s'equivoqués al escriure-ho amb el que el programa deixaria de funcionar. La manera més fàcil de "forçar" que només pugui ser Foc Aigua i Planta es amb un **enum**. Un enum es crea com si fos una classe i permet enumerar diferents opcions



```
public enum Tipus {
    FOC, AIGUA, PLANTA;
}
```

Ara podem afegir una variable de classe Tipus a Mokepon, i modificar els constructors

```
//podem crear múltiples constructors sempre que tinguin paràmetres diferents

public Mokepon(String nom, Tipus tipus) {

    //this.nom fa referència a la variable global i nom a la local que hem passat
    per paràmetre
}
```

```
this.nom = nom;  
this.nivell = 1;  
this.atk = 1;  
this.def = 1;  
this.vel = 1;  
this.tipus = tipus;  
  
}
```

Així com provar que funciona

```
Mokepon marmander = new Mokepon("Marmander", Tipus.FOC);  
  
System.out.println(marmander.nom+" "+marmander.atk);  
  
if(marmander.tipus == Tipus.FOC) {  
    System.out.println("Marmander es de Foc!");  
}
```

FINS AQUI



Atacs

Ara et toca a tu crear una classe. Crea una classe Atac amb els següents atributs:

String nom
double poder (un double entre 10 i 100)
Tipus tipus
int moviments_maxims
int moviments_actuais

Crea un parell de constructors.

Al primer li passes nom, poder, tipus, i moviments_maxims (els actuals els posa igual als maxims)

 Institut Sabadell	UF4. Programació Orientada a Objectes	 Curs: 2020/2021
	<i>Moképon - Part 1</i>	

El constructor verificarà que poder sigui més gran o igual que 10 (si no ho és, ho posarà automàticament a 10) i més petit o igual que 100 (si no ho és, ho posarà automàticament a 100).

L'altre li passes nom i tipus, i posa poder a 10 i moviments_maxims a 10.

Fes que Mokepon tingui una variable atacs que sigui una llista de 2 Atac. Crea un nou mètode a Mokepon que es digui "public void afegirAtac(Atac at)" que afegeixi un nou atac a la llista d'atacs (sempre que en tingui menys de 2). A la classe test, crea dos atacs i afegeix-los a un Mokepon.

Lluita!

Ja estas llest per fer una lluita de Mokepon.

La fórmula per lluitar és la següent

$$Damage = \left(\frac{\left(\frac{2 \times Level}{5} + 2 \right) \times Power \times A/D}{50} + 2 \right) \times Modifier$$

a on Level es el nivell del Mokepon atacant, Power el poder de l'atac, A l'atac del Mokepon atacant, D la defensa del Mokepon que rep l'atac, i Modifier el modificador de tipus, a on és 2 si és efectiu, 1 si es normal, i 0.5 si és poc efectiu



Foc és efectiu contra planta i poc efectiu contra aigua

Planta és efectiva contra aigua i poc efectiva contra foc

Aigua és efectiva contra foc i poc efectiva contra planta.

Pots fer servir aquesta per calcular l'efectivitat. Posa'l com a mètode de Mokepon

```
public double efectivitat(Tipus atac, Tipus defensa) {
    if(atac == Tipus.FOC && defensa == Tipus.AIGUA || atac == Tipus.AIGUA &&
defensa == Tipus.PLANTA || atac == Tipus.PLANTA && defensa == Tipus.FOC ) {
        return 0.5;
    } else if (atac == Tipus.AIGUA && defensa == Tipus.FOC || atac == Tipus.FOC &&
defensa == Tipus.PLANTA || atac == Tipus.PLANTA && defensa == Tipus.AIGUA ) {
```

 Institut Sabadell	UF4. Programació Orientada a Objectes	 Curs: 2020/2021
	<i>Moképon - Part 1</i>	

```

        return 2;

    }else {

        return 1;

    }

}

```

Fes un mètode a Mokepon que sigui public void atacar(Mokepon atacat, int num_atac) a on agafarà l'atac en aquella posició a la llista, calculara el dany segons la fórmula esmentada, i restarà la vida (hp) al Mokepon atacat.

Pensa que la fórmula tornarà un double, així que hauràs de fer un cast a enter per a poder-ho restar de l'hp.

Crea un atribut boolean debilitat i fes també un mètode debilitarse() que activi l'atribut debilitat. Modifica atacar de forma que un Mokepon debilitat no pugui atacar.

Fes per últim un mètode curar() que desactiva debilitat (si estava activat) i posi la vida a hp_max. Assegura't que tots els mètodes son public

Prova la lluita amb el Mokepon al qual li has afegit dos atacs a la classe Test.