Kenan de Vries, studentnummer: 1811470.

2VB

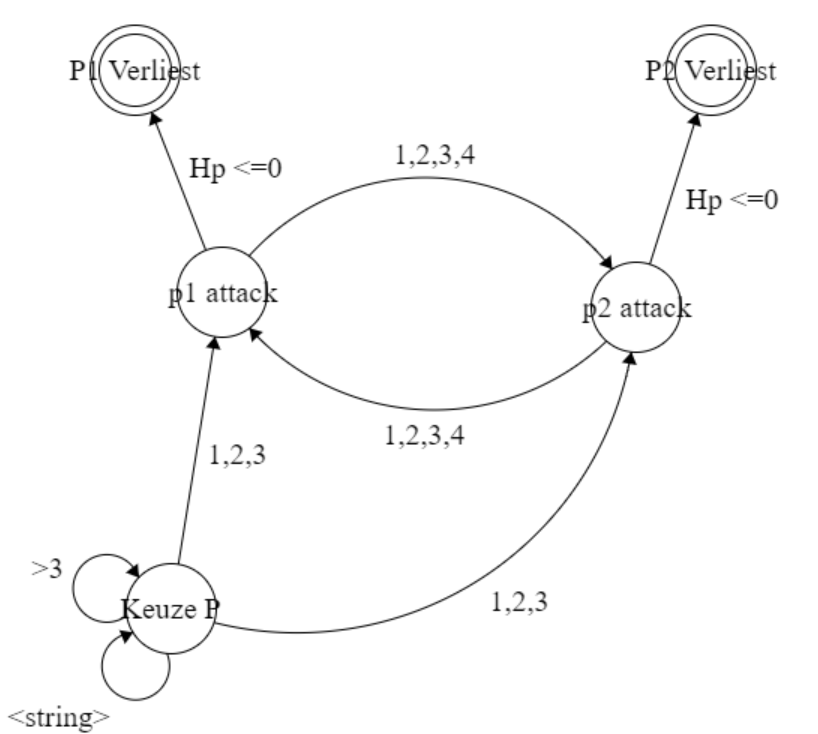
25/11/2022

***Verslag Simulation 1***

**Inleiding:**

In opdracht van Hogeschool Utrecht ben ik aan de slag gegaan met een vereenvoudigde simulatie van een Pokemon gevecht. Hoewel er veel functionaliteit mist ten opzichte van een echte Pokemon game, is het programma voldoende om aan de opdracht van het vak te voldoen. Dit is namelijk een Finite State Machine bouwen en deze als basis gebruiken voor de implementatie van een gekozen casus. De code is gedeeltelijk OOP (Object Oriented Programming) en gedeeltelijk procedureel geprogrammeerd en maakt gebruik van automatentheorie. Dit is jammer genoeg geen document over de gehele code maar er zullen codesnippets worden getoond om een paar keuzes van de implementatie toe te lichten en te verduidelijken.

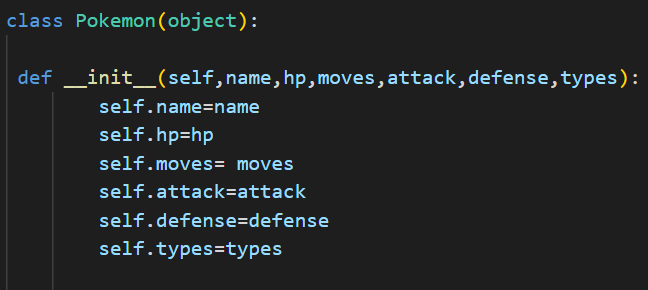
**De Finite State Machine (FSM):**



De Finite State Machine is heel simpel gezegd de blauwdruk van het programma. Het laat de huidige staat van het programma en de input of condities die nodig zijn om deze te veranderen zien.

**De Pokemon class:**

Een gevecht start met 2 pokemons. Deze Pokemons hebben attributen zoals een naam, aanvalsopties en verdedigingspunten etc. Nodig. Hiervoor is een class genaamd (heel verassend) Pokemon aangemaakt. Dit maakt het mogelijk eenvoudig pokemons te constructen. Er is voor nu gekozen om maar 5 pokemons in het spel te doen.

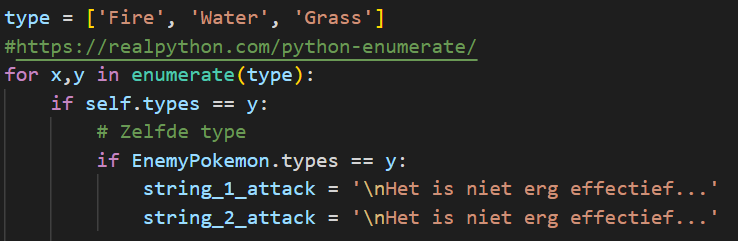


Met een kort voorbeeld van het gebruik:

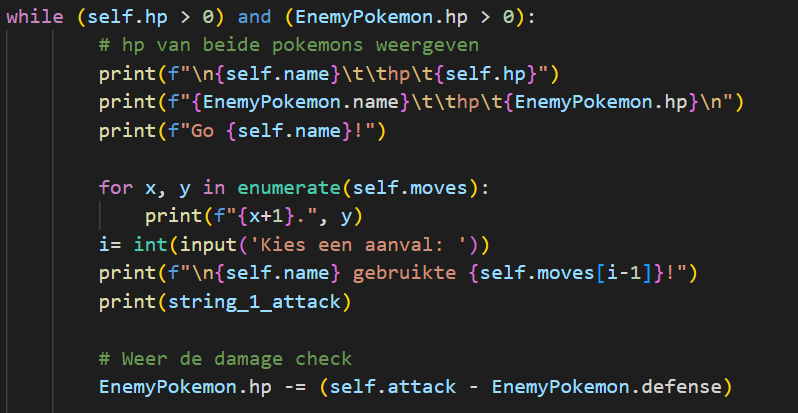


**Functie Def Battle() :**

De daadwerkelijke gevechtslogica zit in de def battle functie van het programma. Dingen als critical hits, special attacks etc. Zijn weggelaten in deze implementatie net als bijna alle Pokemon types. De types die er wél zijn, zijn fire, water en grass. Dit is belangrijk want in Pokemon hebben alle types sterkte en zwaktes voor de andere types. Het vuur type neemt bijvoorbeeld extra veel schade en doet minder schade tegen het water type. Als de types gelijk zijn doen ze beide minder schade tegen elkaar. Omdat het om maar 3 types gaat is dit bijgehouden in een for loop met simpele steen papier schaar logica.



Het gevecht is turn based (de pokemon die als eerste in de code staat gaat ook altijd als eerste) en blijft doorgang zolang beide pokemons boven de 0 zitten. Iedere pokemon heeft 4 mogelijke aanvallen van zijn type. Omdat al vooraf wordt bepaald of de aanval sterk of zwak is tegen het type van de andere pokemon kunnen we dit gedeelte wederom doen met een simpele for loop waardoor die met een index door een lijst van de moves gaat gebaseerd op user input.



**Discussie:**

Ik denk dat de code voldoende is om aan de opdracht te voldoen. De flow van het programma en de link naar een Finite state machine is makkelijk te maken. Echter vind ik het teleurstellend dat het programma zo kaal is vergeleken met een echt pokemon gevecht. Ook had er meer gefocust kunnen worden op OOP programmeren in plaats van de huidige mix, dit had waarschijnlijk onder andere een betere structuur opgeleverd zeker bij uitbreiding van deze versie.