MŰEGYETEM 1782

A PROGRAMOZÁS ALAPJAI 2.

HÁZI FELADAT DOKUMENTÁCIÓ

OOP POKEMON JÁTÉK

KÉSZÍTETTE: KOREN ZOLTÁN , HOTQM1
korenzol.007@gmail.com

KÉSZÍTÉS FÉLÉVE: 2018/19/2

TARTALOMJEGYZÉK

Felhasználói dokumentáció

Elindítás után, egy nevet kér be a program, ennek egyrészt az a célja, hogy megalkossa a felhasználó karakterét a program másrészt az, hogy semmi se kezdődjön el/semmi se történjen a programban, míg az nem szükséges (azaz amíg a felhasználó nem figyel oda vagy bármi egyéb indok). Az alkalmazás használatához a billentyűzet egyes billentyűin kívül nincs szükség. Az alap navigációban/menüben a használható utasításoknak a gombkiosztása a következő: „H" – A pokemonjaink meggyógyítása. „P" – A pokemonjaink kilistázása. „R" – Alapjáraton a Right-ot szimbolizálná, de igazából csak a történetben, az adott részhez tartozó döntésünknél az egyik opciót jelöli. „L" – az előzőhöz kapcsolódóan ez a Left-et szimolizálja, és ugyanaz a működési elve, mint az R-nek. „X"-el tudunk kilépni, ahol egy további megerősítést vár a program, ahol még visszavonhatjuk döntésünket az „N" segítségével. Ennél a résznél az elvárt működés és kimenet az, hogy a megfelelő utasításokhoz illően, a megfelelő dolog történjen storyban. Pl.: Ha beakarunk menni egy házba, akkor az legyen a következő szöveg, hogy már bent vagyunk. Továbbá elvárt működés még az, hogy hogy ha gyógyítjuk a pokemonjainkat akkor az meg is történjen (továbbá ne lehessen olyan pokemont gyógyítani, akinek már így is van elég hp-ja/már nem él), ha azt kérjük, hogy listázza ki a program a pokemonjainkat, akkor standard outputra kerüljön is ki. További része a programnak a harcrendszer. Itt a következő utasítások jöhetnek szóba. „M"-el tudjuk kiválasztani, hogy milyen képességgel akarunk támadni, itt a harcok kiértékelése a háttérben történik meg, a user csak az eredményt láthatja. „S"-el tudunk váltani pokemonjaink között, elvárt működés az, hogy olyan pokemont ne tudjunk választani, amelyik nem él/nincs harcra kész állapotban. „H"-val ugyanazt tudjuk, mint a menüpont esetén. „R"-el meglehet próbálkozni a harctól való meneküléstől (spoiler: nem lehet elmenekülni, ezek itt mind élet-halál küzdelmek),"C"-vel amennyiben egy Pokemon az ellenfelünk, megpróbálhatjuk azt elkapni, ezzel használhatóvá válhat számunkra és feltudjuk használni egy jövőbeli harcban.

Osztályok statikus leírása

Creature

Felelőssége

Egy lényt (pontosabban: pokemon evő szörnyet) megvalósító osztály. Jellegzetessége, hogy ha harcok során nagyon szerencsésen támad, megtudja magát gyógyítani, hisz kiharap egy falatot ellenfeléből

Ősosztályok

Nincs őse, viszont belőle származik a Pokemon osztály.

Attribútumok

Privát

- double hp – A lénynek van életereje.
- string name-A lénynek van neve
- double basedmg-A lénynek megvan szabva, hogy alapjáraton mennyire erős-mennyit sebez
- int maxhp-Illetve meg van neki adva, hogy maximálisan mennyi lehet az életereje, hogy ha gyógyítjuk, akkor ne legyen több életereje, mint újszülött korában.

Védett

-

Publikus

-

Metódusok

Privát

-

Védett

-

Publikus

- Creature() – default konstruktor, egy alapértelmezett állapotot teremt elő
- Creature(double hp, string name, double basedmg, int maxhp) – paraméteres konstruktor
- ~Creature() – destruktor
- string getName()const – visszadja a lény nevét
- double getHp()const – visszadja a lény életerejét
- double getBaseDmg()const – visszadja a lény sebzését
- int getMaxHp()const – visszadja a lény maxéleterejét
- void setBaseDmg(double a)-beállítja a sebzését
- void setMaxHp(int a)-beállítja max életerejét
- void setName(string a) – beállítja a nevét
- virtual void Print()-kiírja adattagjainak értékét(pontosabban hp-t, name-t)
- bool isAlive()-visszatérési értéke 1, ha a lény életereje nagyobb, mint 0
- void setHp(double newhp)-paraméterben adott értékre állítja a lény életerejét, de a maxhpnál soha nem lesz nagyobb életereje
- virtual double attack(Pokemon & enemy)-Mivel Creature csak Pokemon ellen harcolhat, így ez a függvény értékeli ki, hogy konkrétan mennyit is fog belé sebezni
- virtual Creature& operator=(const Creature & other)- az egyenlőség jel operátor túlterhelése, hogy amikor létrehozok new-al egy tömböt, akkor a default konstruktorral megalkotott objektumokat fölül tudjam írni azzal, amivel akarom

Pokemon

Felelőssége

Egy Pokemont modellező osztály, mely ugye egyszerre lény is. Nagyrészt úgy viselkedik, mint egy sima lény, viszont a harcok során saját képességeit használja, hogy legyűrje az ellenfelet.

Ősosztályok

Creatureből származik publikusan. Az öröklődés egyrészt amiatt történik, mert alapjáraton megegyező tagváltozókkal rendelkeznek (hisz pl.: a Pokemonnak is van élete), továbbá hasonlóan is viselkednek, hisz mindkettő tud pl. támadni (az más kérdés, hogy máshogy értékelődnek ki).

Attribútumok

Privát

- Minden, ami a Creature-nek is tagváltozója volt, hisz belőle örkölődik.
- Type own – egy pokemonnak meg van adva, hogy milyen típusú (pl.: Víz, Tűz, stb.)
- Type weakness- egy pokemonnak meg van adva, hogy milyen típusú ellenfelek ellen kevésbé hatékonyabb (pl.: Tűznek a gyengesége a Víz)
- int lvl – egy pokemonnak meg van adva, hogy milyen szintű, mint a régi pokemon játékokban, hisz nagyobb szintű pokemon nagyobbat is sebez

- int growth – ha ez a szám eléri a 100-at, akkor történik szintlépés, tehát kb. azt mutatja, hogy eddig mennyi XP-t szerzett a pokemon, hogy szintet léphessen
- Move * ability – Egy olyan tömb, mely a pokemonnak a képességeit tartalmazza. pl: Mudnip nevezetű pokemonnak van egy Surf nevezetű képessége – az benne van ebben a tömben, hogy tudja azt használni
- int abilitycount – a tömbhöz kapcsolódóan megmondja, hogy mennyi képessége van a pokemonnak

Védett

-

Publikus

-

Metódusok

Privát

-

Védett

-

Publikus

- Pokemon() – default konstruktor, alapértelmezett tulajdonságokkal ruházza fel
- Pokemon(double hp, string name, double basedmg, int maxhp, Type own, Type weakness, int lvl=1, int growth=0) – paraméteres konstruktor, megadott paraméterrel létrehoz egy Pokemon típusú objektumot (ilyenkor még a képességeit tartalmazó tömb nullptr)
- ~Pokemon() – destuktor, felszabadítja a lefoglalt területetet
- Type getWeakness()const – visszadja a pokemon gyengességét(típusát)
- Type getOwn()const – visszadja a pokemon saját típusát
- int getcount()const – visszadja, hogy mennyi képessége van a pokemonnak
- Move getAbility(int idx)const – adott indexű képességet adja vissza (a képességet tömbjéből)
- int getlvl()const – visszaadja, hogy hányas szintű a pokemon
- void Print() – kiírja standard outputra a pokemon tulajdonságait
- bool lvlup(int newgrowth) – megnézi, hogy történik e szintlépés, ha igen akkor lvl++, ha nem akkor nem változik
- void listAbility()- kiírja, hogy milyen képességei vannak a Pokemonnak
- double attack(Pokemon & enemy) – amennyiben ez egy olyan pokemon, mely ellenfelünk lesz a program futása során, úgy ez a metódus fogja kiértékelni, hogy mennyit fog pokemonunkba sebezni.
- void linkAbility(Move & Ability) – hozzá csatol egy új képességet a már meglévő tömbünkhöz
- Pokemon& operator=(const Pokemon & other) - = operátor túlterhelése, hogy a new-al létrehozott pokemon tömbünkben felül tudjuk írni a default konstruktorral készült objektumokat

Move

Felelőssége

Egy képességet modellező osztály; képessége Pokemonnak lehet. Ezek alapján tudjuk kiértékelni a Pokemonoknak a sebzését, továbbá ennek az osztály segítségével tudunk minden egyes Pokemon objektumnak akár teljesen saját képességet adni, ennek az osztálynak segítségével van az, hogy harcok során szabad kezet kap a felhasználó, hogy milyen képességét akarja használni a pokemonnak.

Ősosztályok

Nincs őse, se leszármazottja.

Attribútumok

Privát

- Type own – A képességeknek is van típusa (jellemzően megegyezik a Pokemon típusával, de lehet semleges is.
- string name – Neve is van a képességnek
- double succfaktor – sebzés kiértékelése után, ha éppen kedvező esetet nézünk ezzel beszorzódik a sebzés
- double failfaktor – sebzés kiértékelése után, ha éppen kedvezőtlen esetet nézünk ezzel beszorzódik a sebzés

Védett

-

Publikus

-

Metódusok

Privát

-

Védett

-

Publikus

- Move() – alapértelmezett konstruktor, alapértelmezett értékeket állít be
- Move(Type own, string name, double succfaktor=1.25, double failfaktor=0) – paraméteres konstruktor
- ~Move() – desturktor, felszabadítja a lefoglalt memóriaterületet
- Move(const Move& other)=default – az alapértelmezett másoló konstruktor, magamnak jelzem ezzel, hogy nem kell foglakoznom vele
- string getName() – visszadja a képesség nevét
- double use(Pokemon & owner, Pokemon & enemy) – ha a képességet egy Pokemonon használjuk, ez a függvény hívódik meg
- double use(Pokemon & owner, Creature & enemy) – ha a képességet egy Creature-n használjuk(ami nem Pokemon!), ez a függvény hívódik meg.
- void Print() – adatok kiírására szolgál

Trainer

Felelőssége

Egy Pokemon Trainert modellező osztály. A userunk is egy ilyen Trainer osztályon keresztül tudja használni a programot. Feladata, hogy pokemonokat tároljon, és tudjon harcolni lények ellen. További feladatai a metódusokból eredeztetik.

Ősosztályok

Nincs őse, se leszármazottja.

Attribútumok

Privát

- string name – Hiszen van neve a Trainernek is.
- int pokemoncount – megmondja, mennyi pokemon van eddig nálunk
- Pokemon * pokemons – pokemonokat tartalmazó tömb
- int balls – labdák száma (hiszen szerezhetünk új pokemonokat is!)
- int selectedIndex – jelenleg kiválasztott pokemon indexe (a tömbben)
- int potions – gyógyitalok száma, hiszen pokemonjaink megsérülhetnek a harcok során, lehetőségünk van meggyógyítani őket

Védett

-

Publikus

-

Metódusok

Privát

-

Védett

-

Publikus

- Trainer() – default konstruktor, alapértelmezett adatokkal konstruálja meg az objektumot
- Trainer(string name, int balls, int potions, int pokemoncount, int selecectedIndex=0) – paraméteres konstuktor
- ~Trainer() – destruktor, felszabadítja a lefoglalt területet
- void Print() – kiírja standard outputra az objektum adatait
- bool catchPokemon(Pokemon & newpokemon) – értéke 1, ha a paraméteren adott pokemont sikerül a gyűjteményünkbe csatolni, 0, ha nem sikerül valamilyen oknál fogva)
- void switchtoIndex(int index) – tulajdonképpen egy setter, de kicsit másképp működik, hogy a usernek ne kelljen 0 indexű bármit látnia consolon
- void usePotion() – jelenleg kiválasztott pokemon gyógyítása
- void usePotion(int a) – paraméteren kapott sorszámú pokemon gyógyítása
- void linktoArray(Pokemon & newpokemon) – pokemons tömbhöz csatolja a paraméteren kapott pokemont
- void list() – kilistázza a pokemons tömb tartalmát
- void setBalls(int count) – beállítja a labdák számát
- void setPotions(int count) – beállítja a gyógyitalok számát
- void battle(creature & enemy) – A user által irányított harc függvénye, amennyiben egy sima lénnyel találjuk magunkat szemben
- void battle(Pokemon & enemy) – A user által irányított harc függvénye, amennyiben egy olyan lénnyel találjuk magunkat szemben, aki Pokemon is ráadásul
- bool alivePokemon() – értéke 1, ha még van használható pokemon tömbünkben
- bool curentisUsable() – értéke 1, ha a jelenleg kiválasztott pokemon használható
- Pokemon * getpokemons() – visszaadja a pokemons tömböt

- Trainer& operator=(const Trainer & other) - = operátor túlterhelése, hogy a new-al lefoglalt Trainer típusú tömb elemit felül tudjuk írni.

Story

Felelőssége

A „térkép” alkotást szolgáló osztály, amely azt a célt szolgálja, hogy a program futása során lehessen mászkálni helyekre, döntéseket hozni.

Ősosztályok

Nincs őse, se leszármazottja.

Attribútumok

Privát

- int id – egy azonosító mely megkülönbözteti az egyes objektumokat
- char text[4000] – a szöveg mely majd kiíródik standard outputra (konkrétan a storyt magát foglalja magába)
- int nextright – amennyiben a választás R volt, erre az id-jű elemre lépünk át
- int nextleft – amennyiben a választás L volt, erre az id-jű elemre lépünk át
- static int wildstart – megadja, hogy mikortól kezdődik az a rész a történetben, ahol véletlenszerűen találkozhatunk pokemonokkal
- static int arenastart – megadja, hogy mikortól kezdődik az a rész a történetben, ahol az arena ellenfelekket fogunk találkozni
- static int elementnum – megadja, hogy mennyi Story objektum létezik (tehát, hogy a Story-k tömbje hány elemű)

Védett

-

Publikus

-

Metódusok

Privát

-

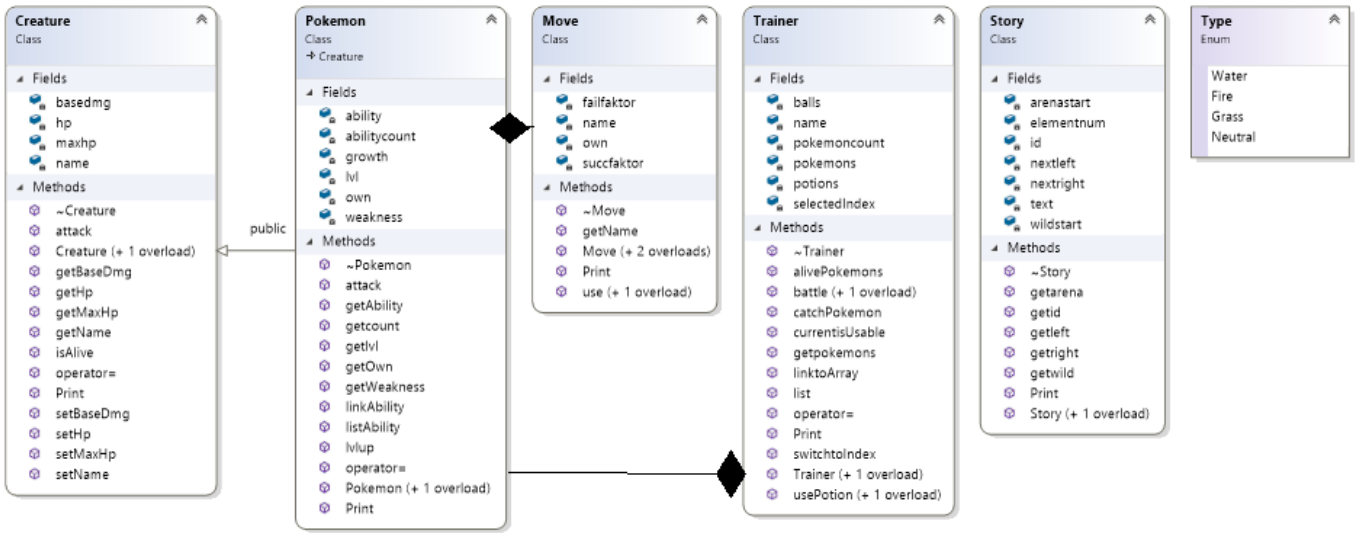
Védett

-

Publikus

- Story() – default konstruktor
- Story(int id, int nextright, int nextleft, char * szoveg) – paraméteres konstruktor
- ~Story() – destruktor, mely felszabadítja a memóriaterületeket
- void Print() – kiírja a textet standart outputra
- int getright()const – visszaadja a nextright értékét
- int getleft()const - visszaadja a nextleft értékét
- int getarena()const - visszaadja az arenastart értékét
- int getwild()const - visszaadja a wildstart értékét
- int getid()const - visszaadja az id értékét

UML osztálydiagramm



Összegzés

Mit sikerült és mit nem sikerült megvalósítani a specifikációból?

Így visszaolvasván a specifikációt, szerintem mindent sikerült megvalósítani, talán az nem sikerült, hogy a githubon lévő projektet rendszerint frissítsem, de viszonylag így is jól követhető a program alakulása. A harcrendszer a tesztek alapján jól működik, a léptetés is hibakezeltem, próbáltam a programomat úgy megírni, hogy teljesen fool-proof legyen, tehát, ha rossz egy bemenet, akkor vagy kérjen egy újat vagy lépjen ki és ne írja tele a console ablakot minden oda nem illő dologgal. Ami talán nem úgy sikerült, ahogy akartam, az a történet. Ugyan nem tartozik a programozás témakörébe, de szerettem volna egy igényes storyt létrehozni, de félév végi hajrá és időhiány miatt, egy nagyon rövid, de tesztelésre alkalmas kis keretet csináltam neki. A pokemonokat tényleg el lehet kapni, aminek nagyon örültem, lehet velük is harcolni.

Mit tanultál a megvalósítás során?

A legnagyobb nehézség az volt, hogy kitaláljam, pontosan mit is szeretnék csinálni. Mivel nem volt konkrét feladat kiírás, hanem saját ötletet akartam megvalósítani, így nehéz volt a fejemben összerakni, hogy mit akarok majd látni kb. ilyenkor a félévben a programom kimenetén. A már említett fool-proof tervezés is sok nehézséget hozott magával, hiszen olyan hibák jöhettek létre a programomban, amikkel előadáson/laboron egyáltalán nem találkozhattam. Egyébként dióhéjban összefoglalva azokat a tanulságokat vonhatom le, hogy:

[1]-Mikor valamilyen feladatot akarunk megoldani, vagy bármit leprogramozni, akkor a fejünkben meg kell már lennie egy váznak, hogy hogyan is akarjuk megoldani azt, mert útközben átírni a kódot nem igazán kellemes időtöltés.

[2]-A teljesen legapróbb funkciók is elég nagy hibákat tud okozni.

A kérdésre meg konkrétan válaszolva azt tudom mondani, hogy megtanultam a c++-os filekezelést, hogyan működik az objektum-orientált programozás (olyan szinten, ahogyan az itt elvárt reményeim szerint)

Továbbfejlesztési lehetőségek

Sokat lehetne még rajta fejleszteni. Például lehetne Poke-Centereket is belerakni a játékba, hogy ott pénzért lehessen vásárolni pokelabdákat, gyógyitalokat, lehetne grafikus felületet is csinálni neki, nagyobb térképet, rendesebb storyt, a pokemonok fejlődéskor átmennének egy olyan evolúción, mint ahogyan az a rendes pokemon játékokban is végigmegy, továbbá el lehetne menekülni harc közben, több pokemonnal lehetne egyszerre harcolni, stb. Tehát konkrétan a lehetőség megvan bőven, hiszen magát ezt a programot is a rendes pokemon játék mintájára készítettem el.

Képernyőképek a futó alkalmazásról

```
Welcome! Type your name: Prog2HF
Pokemon name    HP      Status
[1] - Mudkip    25      Usable
[2] - Charman   20      Usable
[3] - Lotad     22      Usable
       Player Instructions: If you want to go left, type "L". If you want to go right, type "R". If you want to heal yo
ur Pokemons, type "H". If you want to exit, type "X". During your story, there might be encounters. Keep in mind that af
ter your battle is over, you will still have to react (type something) to the latest thing I (the narrator) told you. An
yway... Type (R) or (L) to continue...
Choices: (R)IGHT, (L)EFT, (H)EAL, (P)okemonlist E(X)IT
```

```
       You live in a peaceful village called Shelbyville. Your family is well-known for it's bravery and for it's long-
long history of dealing with the ancient creatures, called Pokemons. One day (to be extremely precise: today), you woke
up to the sound of unimaginable horror. Your village was on fire, the cattle were slaughtered, and the only thing you fo
und was your bag with your favourite pokemons in it, and a note on it's side. "We have your family. Come attend the Aren
a in Dhirim, and you might find your loved ones alive". However you know, that the road dangerous, and since they didn't
 take your nintendo, you might aswell just lay back, relax, play a better Pokemon game or something. (R) To help your fa
mily, and go in a dangerous mission to reclaim what is yours or (L) to do what you always do in your life: lose.
...
Choices: (R)IGHT, (L)EFT, (H)EAL, (P)okemonlist E(X)IT
R
       There are several ways to get to Dhirim, but the most ambitious one is through the Calico Desert. It is well-kno
wn for it's pokemon-eating monsters, and thus it is well-known for it's population aswell, which is approximatelly zero.
 However, there is another road, through the Elwynn Forest, which has no pokemon eating monsters but pokemons, who are a
lso dangerous. You decide to: (R) Go through the desert (L) Go through the forest.
...
Choices: (R)IGHT, (L)EFT, (H)EAL, (P)okemonlist E(X)IT
```

```
Cool battle-music playing
The battle take place here and now.
Your pokemon: Name:Mudkip       Healthpoints:25 Current level: 1        Type:1
Your enemy: Name:Szamtud        Healthpoints:20
------------------It's your turn!. H - Heal, M - Moves, S - Switch Pokemons, R - Retreat-----------
M
[1] - Surf
[2] - Tornado
[3] - Punch
Index of desired move:
2
Mudkip is using Tornado.
Mudkipcauses 7.7damage on Szamtud
It's effective

-----------------------------------It's your enemys turn!-----------------------------------
ENEMY:Attack is regular!
Damage given 5
Your pokemon: Name:Mudkip       Healthpoints:20 Current level: 1        Type:1
Your enemy: Name:Szamtud        Healthpoints:12.3
------------------It's your turn!. H - Heal, M - Moves, S - Switch Pokemons, R - Retreat-----------
H
Can't heal the selected pokemon


-----------------------------------It's your enemys turn!-----------------------------------
ENEMY:Attack is not that effective!
Damage given 3.75
Your pokemon: Name:Mudkip       Healthpoints:16.25      Current level: 1        Type:1
Your enemy: Name:Szamtud        Healthpoints:12.3
------------------It's your turn!. H - Heal, M - Moves, S - Switch Pokemons, R - Retreat-----------
```