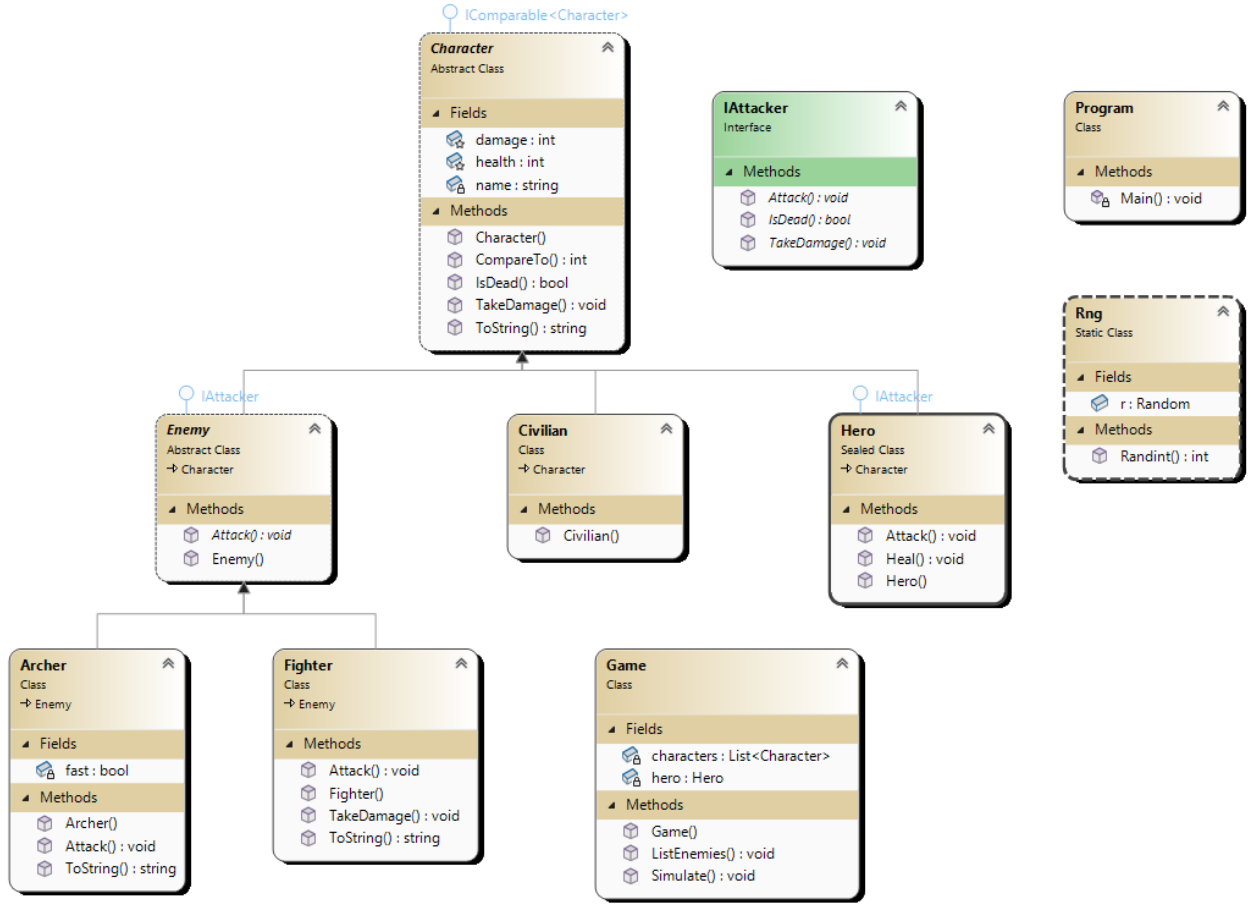
Game – öröklődés gyakorló

Készíts egy játék szimulációt, amelyben egy hős küzd meg ellenfelekkel! Az osztályok adattagjait, metódusait és azok láthatóságát az alábbi osztály diagram alapján vedd fel!



A feladatok csak a diagramról nem leolvasható részleteket tartalmazzák. Érdemes a feladatok sorrendjében haladva kialakítani az osztály hierarchiát.

Rng osztály:

1. Az **Rng** egy statikus osztály[1], melynek `Randint(a, b)` statikus metódusa ad egy véletlen számot az `[a..b]` intervallumról, beleértve a határokat is.

Character osztály:

2. Egy karakter „`{név} (DMG: {sebzés} HP: {életerő})`” formában jelenjen meg szövegként!

   Pl.: `Legolas (DMG: 8 HP: 65)`

[1] Nem példányosítható.

3. A `TakeDamage(int amount)` metódus paraméterként kapja a sebződés mértékét, majd ennek megfelelően csökkenti a karakter életerejét. Ügyelj rá, hogy az életerő ne lehessen negatív és alosztályokból a metódust felül tudjuk írni később!

4. Az `IsDead()` metódus akkor ad igaz értéket, ha a karakter halott, vagyis életereje 0.

5. A **`Character`** osztály implementálja az **`IComparable<Character>`** interfészt, vagyis tudjunk karaktereket egymással összehasonlítani. Azt a karaktert tekintjük kisebbnek (gyengébbnek), akinek a sebzésének és életerejének a szorzata kisebb!

Civilian osztály:

6. A civilek mindannyian 0 sebzéssel és 10 életerővel rendelkeznek. A **`Civilian`** osztály konstruktora csak egy paraméteres legyen, a nevet kapja meg paraméterként!

IAttacker interfész:

7. Az `Attack(Character target)` metódus paramétere egy karakter objektum (a támadás célpontja).

8. A `TakeDamage(int amount)` paramétere a sebződés mértéke.

Hero osztály:

9. A **`Hero`** osztály a **`Character`** osztály leszármazottja, és implementálja az **`IAttacker`** interfészt. A hős támadásakor ellenfele a `damage` adattag értékével sebződik. Használd fel a **`Character`** osztály `TakeDamage()` metódusát!

10. Egy hős gyógyulása során az aktuális életereje 20%-kal növekszik.

11. Oldd meg, hogy a **`Hero`** osztályból ne lehessen leszármaztatni további osztályokat!

Enemy osztály:

12. Az **`Enemy`** osztály őse szintén a **`Character`**. Egy ellenség is tudjon támadni, tehát valósítsa meg az **`IAttacker`** interfészt.

    Az `Attack()` metódusát az interfésznek absztraktként add meg, a konkrét implementáció majd a leszármazott osztályokba fog kerülni.

Archer osztály:

13. Az **`Archer`** osztály konstruktorban fogadja további paraméterként a logikai típusú `fast` adattag értékét, ami azt jelöli, hogy gyors-e az íjász.

14. Egy normál íjász támadásakor egyszer sebzi ellenfelét, de egy gyors íjász egymás után 3-szor is.

15. Íjászok szöveges megjelenítésekor az ősosztálytól örökölt `ToString()` metódus értéke előtt jelenjen meg a minta szerinti „`[Archer]::`” felirat, és egy szóköz is!

Fighter osztály:

16. Egy harcos csak egyszer tudja sebezni ellenfelét az `Attack()` metódusában. (Tehát ugyanaz a függvénytörzs, ami a **`Hero`** osztály esetében volt.)

17. Az íjászokhoz hasonló módon írd felül a `ToString()` metódust, hogy a sor elején jelenjen meg a „`[Fighter]::`” szöveg, majd egy szóköz a minta szerint!

18. Egy harcosnak van pajzsa, amivel tud védekezni. A `TakeDamage()` metódusban sorsolj ki véletlenszerűen (az **`Rng`** statikus osztály használatával) egy 0 és 10 közé eső számot. Az életerőt ezután ne a támadás mértékével csökkentsd, hanem vond ki belőle a védekezésként generált random számot. Ügyelj rá, hogy az életerő ne növekedhessen a metódusban.

    Pl.: ha 13-mas a támadási érték, amelyhez generálunk egy 5-ös védekezési értéket, akkor 8 sebzést szenved el a harcos, tehát ennyivel csökken az életereje.

    Pl.: ha 5-ös támadáshoz generálunk egy 7-es védekezési értéket, akkor a harcos életereje nem változik, kivédte a támadást.

Game osztály:

19. A **`Game`** osztály konstruktora egy fájl nevét fogadja, amely a következő felépítésű:

```
Archer Legolas 65 8 gyors
Fighter Koponyatörő 130 22
Fighter Ragnar 80 15
Fighter Bjorn 95 18
Fighter Thorvald 120 20
Archer Sólyomszem 55 10
Archer Nyílmester 40 10 gyors
Archer FeketeÍjász 70 14
...
```

    Az egyes sorok adatai szóközzel tagoltak. Az első szó minden esetben az ellenség típusa, vagyis **`Archer`** vagy **`Fighter`**. Ezt követi a neve, az életereje, a sebzése, végül gyors íjászok esetén a sor végén egy „gyors” felirat.

    Olvasd be a fájl tartalmát (hibakezelésre ügyelve), majd hozd létre a megfelelő objektum példányokat, és tárold el a `characters` listában az ellenségeket!

    Szintén a konstruktorban példányosíts egy hőst „Neo” néven, 42 sebzéssel és 100 életerővel.

20. A `ListEnemies()` metódusban rendezd a karaktereket, majd írasd ki őket a konzolra!

21. A hős a rendezett lista szerint megküzd minden ellenséggel. A `Simulate()` metódus feladata a harcok szimulálása. Egy harc addig tart, amíg a hős is él, és az ellenfele is. Elsőként a hős támad, ezután ha ellenfele még él, akkor visszatámad. A harc után kerüljön a konzolra, hogy „Győzelem!” vagy „Vereség!” lett a hős sorsa. Győzelem esetén gyógyuljon a hős, majd folytatódjon a szimuláció a következő küzdelemmel.

Minta konzol:

```
[Archer]:: Vadász (DMG: 7 HP: 45)
[Archer]:: Nyílmester (DMG: 10 HP: 40)
[Archer]:: Legolas (DMG: 8 HP: 65)
[Archer]:: Sólyomszem (DMG: 10 HP: 55)
[Archer]:: Szélvész (DMG: 11 HP: 50)
[Archer]:: FeketeÍjász (DMG: 14 HP: 70)
[Fighter]:: Kardfény (DMG: 13 HP: 90)
[Fighter]:: Ragnar (DMG: 15 HP: 80)
[Fighter]:: Bjorn (DMG: 18 HP: 95)
[Fighter]:: Morgar (DMG: 16 HP: 110)
[Fighter]:: Thorvald (DMG: 20 HP: 120)
[Fighter]:: Koponyatörő (DMG: 22 HP: 130)

[Archer]:: Vadász (DMG: 7 HP: 45)
Győzelem!
[Archer]:: Nyílmester (DMG: 10 HP: 40)
Győzelem!
[Archer]:: Legolas (DMG: 8 HP: 65)
Győzelem!
[Archer]:: Sólyomszem (DMG: 10 HP: 55)
Győzelem!
[Archer]:: Szélvész (DMG: 11 HP: 50)
Győzelem!
[Archer]:: FeketeÍjász (DMG: 14 HP: 70)
Győzelem!
[Fighter]:: Kardfény (DMG: 13 HP: 90)
Győzelem!
[Fighter]:: Ragnar (DMG: 15 HP: 80)
Győzelem!
[Fighter]:: Bjorn (DMG: 18 HP: 95)
Győzelem!
[Fighter]:: Morgar (DMG: 16 HP: 110)
Győzelem!
[Fighter]:: Thorvald (DMG: 20 HP: 120)
Vereség!
```