1. Óra – Egyszerű osztályok készítése

OE.Prog2.Jatek.Jatekter névtéren belül készítsük el az alábbiakat:

Új osztály: **JatekElem**

A *JatekElem* osztályt egészítsük ki az alábbiakkal:

- Új privát mező: x – egész szám, az elem pozíciójának x koordinátája
- Új privát mező: y – egész szám, az elem pozíciójának y koordinátája
- Új publikus tulajdonság: X – visszaadja és módosítja az x értéket.
- Új publikus tulajdonság: Y – visszaadja és módosítja az x értéket.

Új osztály: **JatekTer**

A *JatekTer* osztályt egészítsük ki az alábbiakkal:

- Új konstans: **MAX_ELEMSZAM** – egész szám, a tárolható elemek maximális száma (1000).
- Új privát mező: **elemN** – egész szám, ami azt mutatja, hogy éppen hány elemet tárolunk.
- Új privát mező: **elemek** – egy *MAX_ELEMSZAM* méretű, *JatekElem* objektumokat tartalmazó tömb.
- Új privát mező: **meretX** – egész szám, a játéktér maximális mérete x irányban.
- Új publikus tulajdonság: **MeretX** – csak olvasható, visszaadja a *meretX* értékét.
- Új privát mező: **meretY** – egész szám, a játéktér maximális mérete y irányban.
- Új publikus tulajdonság: **MeretY** – csak olvasható, visszaaadja a *meretY* értékét.
- Új publikus konstruktor: **JatekTer** – két paramétere legyen, amelyek beállítják a *meretX* és *meretY* mezők értékét.
- Új publikus metódus: **Felvetel** – felveszi a paraméterként átadott *JatekElem* típusú objektumot az *elemek* tömbbe. Értelemszerűen növeli az *elemN* mező értékét is.
- Új publikus metódus: **Torles** – törli a paraméterként átadott *JatekElem* objektumot az elemek tömbből (és értelemszerűen csökkenti az *elemN* mező értékét is).
- Új publikus metódus: **MegadottHelyenLevok** – három paramétere van: egy x és y koordináta, illetve egy távolság, a visszatérési értéke pedig egy *JatekElem* objektumokat tartalmazó tömb. Az *elemek* tömb adatai alapján meg kell számolnia, hogy az x és y koordináták által meghatározott ponttól mért *távolság* távolságon belül hány darab *JatekElem* objektum található. Létre kell hoznia egy ekkora *JatekElem* tömböt, majd ebbe ki kell válogatnia az előző feltételnek megfelelő elemeket. Ez lesz a metódus visszatérési értéke.
- Új publikus metódus: **MegadottHelyenLevok** – ez előzőhöz hasonló, de csak x és y paramétere van. Visszatérési értéke egy *JatekElem* tömb, ami azokat az elemeket tartalmazza, amelyek pont az x és y által megadott helyen vannak (célszerű felhasználni az előző metódust, 0 távolsággal).

A már meglévő JatekElem osztályt egészítsük ki az alábbiakkal:

- Új privát mező: **ter** – tárolja annak a *JatekTer* típusú játéktérnek a referenciáját, amelyikbe elhelyeztük.
- Új publikus konstruktor: **JatekElem** – paraméterként megkapja az x és y koordinátákat, illetve egy *JatekTer* referenciát. Mindhárom paramétert mentse el az ezekre szolgáló mezőkbe, majd vegye fel magát a *JatekTer* objektum *Felvetel* metódusával.

Tesztelés: A főprogramban készítsünk egy *JatekTer* objektumot, illetve néhány ebbe elhelyezkedő *JatekElem* objektumot. Próbáljuk ki a megvalósított metódusokat:

- Töröljünk egy játékelemet!
- Adjuk meg, hogy egy megadott helyen épp hány játékelem található!
- Listázzuk egy megadott pont megadott környezetében lévő elemek koordinátáit!

2. Óra – Öröklés és polimorfizmus

OE.Prog2.Jatek.Jatekter névtéren belül készítsük el az alábbiakat:

A már meglévő *JatekElem* osztályt egészítsük ki az alábbiakkal:

- Maga a *JatekElem* osztály legyen absztrakt (az előző órai főprogramot emiatt ki is törölhetjük, mivel nem fog lefordulni).
- A **ter** mező legyen védett láthatósági szintű, hogy a leendő leszármazottak hozzáférhessenek.
- Új absztrakt publikus tulajdonság: **Meret** – lebegőpontos szám, csak olvasható, ezen a szinten még nem tudjuk meghatározni (0..1 közötti értéke lehet majd, egy mezőbe elhelyezkedő elemek mérete nem lehet 1-nél több).
- Új absztrakt publikus metódus: **Utkozes** – paraméterként egy *JatekElem* objektumot kap, ezen a szinten még nem tudjuk meghatározni (ez fogja majd kezelni a különböző típusú elemek ütközéseit).

Új absztrakt osztály: **RogzitettJatekElem**, ami a *JatekElem* leszármazottja

A *RogzitettJatekElem* osztályt egészítsük ki az alábbiakkal:

- Új publikus konstruktor: **RogzitettJatekElem** - paraméterként megkapja az x és y koordinátákat, illetve egy *JatekTer* referenciát. Ezeket továbbítsa az ősének konstruktorához.

Új absztrakt osztály: **MozgoJatekElem**, ami a *JatekElem* leszármazottja

A *MozgoJatekElem* osztályt egészítsük ki az alábbiakkal:

- Új publikus konstruktor: **MozgoJatekElem** - paraméterként megkapja az x és y koordinátákat, illetve egy *JatekTer* referenciát. Ezeket továbbítsa az ősének konstruktorához.
- Új privát mező: **aktiv** – logikai mező, ami azt mutatja majd, hogy működik/él-e még az adott objektum.
- Új publikus tulajdonság: **Aktiv** – visszaadja, illetve felülírja az *aktiv* mező tartalmát.
- Új védett metódus: **AtHelyez** – két egész szám paramétert kap, ujx és ujy. Az ide való átlépés az alábbi műveletekből álljon:
  - A metódus kérdezze le, hogy az új koordináták által megadott helyen éppen milyen más játékelemek találhatók (ez a *JatekTer* segítségével egyszerű).
  - Ezeknek az elemeknek egyesével hívja meg az *Utkozes* metódusát, paraméterként átadva önmagát (tehát a mozgatandó objektum nekiütközik a cél helyen lévőknek).
  - Majd tegye meg ezt fordítva is (tehát a cél helyen lévő ütközik neki a mozgatandó objektumnak).
  - Minden ütközés után ellenőrizze, hogy még *Aktiv*-e a mozgatandó objektum (mivel lehet, hogy valamelyik ütközés hatására meghalt). Ha már nem az, akkor ne folytassa az ütközéseket.
  - Ha az ütközések után még *Aktiv* a mozgatandó objektum, akkor kérje le ismét a cél helyen található elemeket (mivel azok között is lehetett halál az üközések miatt).
  - Számolja ki, hogy mennyi a cél helyen már meglévő elemek összesített mérete (*Meret* tulajdonságok).
  - Ha ehhez még hozzáadjuk az odaléptetni kívánt elem méretét, akkor ez nem haladhatja meg az 1-et. Ha ez alapján elvégezhető a lépés, akkor az objektum *x* és *y* koordinátáit léptessük az új helyre. Ha nem, akkor ne változtassuk meg.

OE.Prog2.Jatek.Szabalyok névtéren belül készítsük el az alábbiakat:

Új osztály: **Fal**, ami a *RogzitettJatekElem* leszármazottja

A *Fal* osztályt egészítsük ki az alábbiakkal:

- Új konstruktor: **Fal** – az őséhez hasonlóan x és y és játéktér paramétereket kap, ezeket továbbítja az ősnek.
- Új publikus felülírt mező: **Meret** – az ősben még absztrakt méretet itt már meg tudjuk határozni. Mindig adjon vissza 1-et (tehát egy fal mellé más már biztos nem fog elférni).
- Új publikus felülírt metódus: **Utkozes** – ne csináljon semmit, a fal passzív.

Új osztály: **Jatekos**, ami a *MozgoJatekElem* leszármazottja

A *Jatekos* osztályt egészítsük ki az alábbiakkal:

- Új privát mező: **nev** – szöveg, tárolja a játékos nevét.
- Új publikus tulajdonság: **Nev** – visszaadja a játékos nevét.
- Új konstruktor: **Jatekos** – négy paramétere van: név, x, y, játéktér. Az elsőt eltárolja a megfelelő mezőben, a többit továbbítja az ős konstruktorának.
- Új publikus felülírt mező: **Meret** – Mindig adjon vissza 0.2-t (tehát 5 játékos fér el egy mezőn, ha nincs ott más).
- Új publikus felülírt metódus: **Utkozes** – ne csináljon semmit.
- Új privát mező: **eletero** – egész szám, kezdőértéke 100. Tárolja a játékos életerejét.
- Új publikus metódus: **Serul** – paramétere egy egész szám. Ha a játékos életereje már eleve 0, akkor ne csináljon semmit. Ha nagyobb, akkor csökkentse az *eletero*-t a paraméterként átadott értékkel (de 0-nál kisebb ne legyen). Ha a játékos életereje elérte a 0-t, akkor az *Aktiv* tulajdonságot állítsa hamisra.
- Új privát mező: **pontszam** – egész szám, kezdőértéke 0. Tárolja, hogy a játékos eddig hány pontot szerzett a játékban.
- Új publikus metódus: **PontotSzerez** – paramétere egy egész szám. A megadott paraméterrel növeli a *pontszam* mező értékét.
- Új publikus metódus: **Megy** – két paramétere két relatív koordináta (rx és ry, tipikusan -1, 0, 1 értékeket várunk ezekben). Számolja ki, hogy az aktuális pozícióból ezzel az elmozdulással hova jutna, és hívja meg az ősétől örökölt *Athelyez* függvényt ezekkel a koordinátákkal.

Új osztály: **Kincs**, ami a *RogzitettJatekElem* leszármazottja

A *Kincs* osztályt egészítsük ki az alábbiakkal:

- Új konstruktor: **Kincs** – az őséhez hasonlóan x és y és játéktér paramétereket kap, ezeket továbbítja az ősnek.
- Új publikus felülírt mező: **Meret** – Mindig adjon vissza 1-et.
- Új publikus felülírt metódus: **Utkozes** – Ez akkor fog lefutni, ha valaki nekiütközött egy kincs objektumnak. Ilyekor a teendők:
  - Ellenőrizzük, hogy a nekiütköző (tehát a paraméterként kapott *JatekElem*) egy *Jatekos* típusú objektum-e.
  - Ha igen, akkor hívjuk meg annak *PontSzerez* metódusát, paraméterként 50-et átadva (tehát a kincsért kapott 50 pontot).
  - A kincs törölje önmagát a játéktérről (a *JatekTer* osztály *Torles* metódusával).

OE.Prog2.Jatek.Keret névtéren belül készítsük el az alábbiakat:

Új osztály: **Keret**

A *Keret* osztályt egészítsük ki az alábbiakkal:

- Új konstans: **PALYA_MERET_X** – a pálya maximális mérete x irányba (21)
- Új konstans: **PALYA_MERET_Y** – a pálya maximális mérete y irányba (11)
- Új konstans: **KINCSEK_SZAMA** – a pályán lévő kincsek száma (10)
- Új privát mező: **ter** – egy *JatekTer* objektum, ami a játékteret fogja tárolni.
- Új privát metódus: **PalyaGeneralas** – hozzon létre és a *ter*-be helyezzen el **PalyaElemeket** az alábbiak szerint:

- A pálya négy szélén legyenek falak (tehát *Fal* objektumok)
- Véletlenszerűen szórjon szét *KINCSEK_SZAMA* darab kincset a pályán belül, ügyelve arra, hogy egy helyre nem fér el egynél több *Kincs* objektum (az 1,1 pozíciót is hagyjuk szabadon, innen indul majd a játékos).

- Új publikus konstruktor: **Keret** – hozza létre a *PalyaTer* objektumot a maximális mérettel, majd hívja meg a *PalyaGeneralas* metódust.
- Új privát mező: **jatekVege** – alapértelmezetten hamis, csak akkor lesz igaz, ha vége a játéknak.
- Új publikus metódus: **Futtatas** – ez fogja elindítani és működtetni az egész játékot. A metódus fő feladatai:
  - Létrehoz egy játékost „Béla” néven az 1,1 pozícióba a játéktérben
  - Egy ciklus addig fusson, amíg a *jatekVege* metódus nem vált igazra. A cikluson belül kérjen be egy billentyűleütést, és ha ez egy kurzor gomb volt, akkor mozgassa Bélát ebbe az irányba, ha az Esc billentyű, akkor pedig lépjen ki.

```
do
{
    ConsoleKeyInfo key = Console.ReadKey(true);
    if (key.Key == ConsoleKey.LeftArrow) jatekos.Megy(-1, 0);
    if (key.Key == ConsoleKey.RightArrow) jatekos.Megy(1, 0);
    if (key.Key == ConsoleKey.UpArrow) jatekos.Megy(0, -1);
    if (key.Key == ConsoleKey.DownArrow) jatekos.Megy(0, 1);
    if (key.Key == ConsoleKey.Escape) jatekVege = true;
} while (!jatekVege);
```

Tesztelés: A főprogramban hozzunk létre egy *Keret* objektumot, és hívjuk meg a *Futtatas* metódusát. Nincs még megjelenítési rétegünk, így nem látunk belőle semmit, de a vezérlés és az üzleti logika már működik, a játék használható (a játékos mozog, kincseket fel lehet venni, nem lehet rámenni a falakra, stb.). Bár határozottan elítéljük az üzleti logikából a konzolra való kiírást, de tesztelési célokból ez elfogadható: egészítsük ki úgy a *Kincs* osztályt, hogy a konstruktorban írja ki a képernyőre az x és y koordinátát (tehát a program indulásakor megjelenik a 10 elkészített kincs helye), illetve az *Utkozes* metódusát úgy, hogy a pont növelésekor írja ki a konzolra a *Jatekos* nevét. Így a megjelenítést helyettesítendő, a kincseket papírra felrajzolva, vakon tudjuk teljesíteni a pályát.

3. Óra – Interfészek (megjelenítési szint)

Töltsük le a **SzalbiztosKonzol.cs** filet, és másoljuk a projectünkhöz. Ennek az osztálynak a statikus metódusaival tudunk a képernyő megadott pozíciójába egy karaktert, vagy egy szöveget kiírni.

OE.Prog2.Jatek.Megjelenites névtéren belül készítsük el az alábbiakat:

Új publikus interfész: **IKirajzolhato**

Az *IKirajzolhato* interfészt egészítsük ki az alábbiakkal:

- Új tulajdonság: **X** – csak olvasható egész. A megjelenítendő alak x koordinátája.
- Új tulajdonság: **Y** – csak olvasható egész. A megjelenítendő alak y koordinátája.
- Új tulajdonság: **Alak** – csak olvasható karakter. A jel, ami megjelenik majd.

Új publikus interfész: **IMegjelenitheto**

Az *IMegjelenitheto* interfészt egészítsük ki az alábbiakkal:

- Új tulajdonság: **MegjelenitendoMeret** – csak olvasható egész tömb. Mindig egy kétdimenziós egész tömböt várunk, ami tartalmazza a megjelenítendő terület szélességét és magasságát.
- Új metódus: **MegjelenítendőElemek** – nincs paramétere, visszaad egy *IKirajzolhato* elemekből álló tömböt.

Új publikus osztály: **KonzolosMegjelenito**

A *KonzolosMegjelenito* osztályt egészítsük ki az alábbiakkal:

- Új privát mező: **forras** – egy *IMegjelenitheto* típusú mező, innen jönnek majd a megjelenítendő adatok.
- Új privát mező: **pozX** – ennyivel kell majd eltolni a kirajzolandó adatok x koordinátáját (ez akkor lesz érdekes, ha több *KonzolosMegjelenito* is dolgozik egyidőben, akkor más-más helyre rajzolják ki az adataikat).
- Új privát mező: **pozY** – ennyivel kell majd eltolni a kirajzolandó adatok y koordinátáját.
- Új publikus konstruktor: **KonzolosMegjelenito** – paraméterként kapott három értékkel feltölti az előző három mezőt.
- Új publikus metódus: **Megjelenites** – kiolvassa a *forras*-ból a kirajzolandó elemeket, majd ezeket kirajzolja a *pozX* és *pozY* által eltolt helyre:
  - A *forras* objektum *MegjelenitendoElemek* metódusával kiolvassa a kirajzolandó elemek tömbjét.
  - Lekérdezi azt is, hogy mekkora a megjelenítendő terület mérete (szintén a forrás adja meg egy kétdimenziós tömbben a szélességet és a magasságot).
  - Ezt követően két egymásba ágyazott ciklussal végigszaladunk a méret által megadott területen, ha
    - a forrás által visszaadott objektumok egyike sincs a megadott ponton, akkor kiírunk oda egy szóközt a *SzalbiztosKonzol* segítségével,
    - a megadott ponton van valami, akkor pedig annak az *Abra* tulajdonsága által visszaadott karaktert írjuk ki a megadott helyre.
  - A megjelenítés során a kirajzolandó elem x és y koordinátáit mindig toljuk el a megjelenítő *pozX* és *pozY* mezőjének értékével.

OE.Prog2.Jatek.Szabalyok névtéren belül készítsük el az alábbiakat:

A *Fal* osztály valósítsa meg az *IKirajzolhato* interfészt az alábbiak szerint:

- Az interfész által igényel *X* és *Y* tulajdonságokat már az ős *JatekElem* osztály megvalósította, így ezeket implicit módon már implementáltuk.
- Új publikus tulajdonság: **Alak** – csak olvasható, visszatérési értéke `'\u2593'`

A *Jatekos* osztály valósítsa meg az *IKirajzolhato* interfészt az alábbiak szerint:

- Az interfész által igényel *X* és *Y* tulajdonságokat már az ős *JatekElem* osztály megvalósította, így ezeket implicit módon már implementáltuk.
- Új publikus tulajdonság: **Alak** – csak olvasható, visszatérési értéke legyen attól függően, hogy ha még aktív, akkor `'\u263A'`, különben `'\u263B'`

A *Kincs* osztály valósítsa meg az *IKirajzolhato* interfészt az alábbiak szerint:

- Az interfész által igényel *X* és *Y* tulajdonságokat már az ős *JatekElem* osztály megvalósította, így ezeket implicit módon már implementáltuk.
- Új publikus tulajdonság: **Alak** – csak olvasható, visszatérési értéke `'\u2666'`

OE.Prog2.Jatek.JatekTer névtéren belül készítsük el az alábbiakat:

A *JatekTer* osztály valósítsa meg az *IMegjelenitheto* interfészt az alábbiak szerint:

- Új publikus tulajdonság: **MegjelenitendoMeret** – azt adja vissza, hogy mekkora területet kell majd megjeleníteni. A tulajdonság hozzon létre egy kétdimenziós egész tömböt, ebbe helyezze el a *meretX* és *meretY* mezők értékét, majd ezt adja vissza.
- Új publikus metódus: **MegjelenitendoElemek** – ez adja vissza azoknak az elemeknek a listáját, amelyeket ki kell rajzolni. Ehhez:
  - Számolja meg, hogy az *elemek* tömbben hány olyan objektum van, aki megvalósítja az *IKirajzolhato* interfészt.

- Hozzon létre egy ekkora, *IKirajzolhato* nevű tömböt **vissza** néven.
- Ebbe a tömbbe válogassa ki az interfészt megvalósító elemeket.
- Majd adja vissza ezt a tömböt visszatérési értékként.

OE.Prog2.Jatek.Keret névtéren belül készítsük el az alábbiakat:

A *Keret* osztályt egészítsük ki az alábbiakkal:

- Meglévő metódus módosítása: **Futtatas** – a metódus elején hozzunk létre egy új *KonzolosMegjelenito* objektumot, paraméterként átadva a *ter* objektumot forrásként, és a 0,0 koordinátákat megjelenítési pozícióként.
- Meglévő metódus módosítása: **Futtatas** – a billentyűleütéseket figyelő ciklust egészítsük ki azzal, hogy minden iterációban meghívja ennek a megjelenítőnek a *Megjelenites* metódusát.

Már működik a megjelenítési szint, az előző órán, a *Kincs* osztályba írt konzol kiírásokat töröljük!

OE.Prog2.Jatek.Szabalyok névtéren belül készítsük el az alábbiakat:

A *Jatekos* osztály valósítsa meg az *IMegjelenitheto* interfészt az alábbiak szerint:

- Új publikus tulajdonság: **MegjelenitendoMeret** – mivel a játékosnak van egy referenciája ahhoz a *JatekTer*-hez, amelyikben van, így adja vissza annak a méreteit.
- Új publikus tulajdonság: **MegjelenitendoElemek** – a játékos a pálya 5 sugarú környezetét látja be, így az itt található elemeket fogja visszaadni. Ehhez:
  - Kérje le a játéktértől az ő 5 sugarú környezetében található elemek tömbjét.
  - Ebben számolja meg, hogy hány olyan objektum van, ami megvalósítja az *IKirajzolhato* interfészt.
  - Hozzon létre egy ekkora, *IKirajzolhato* nevű tömböt **vissza** néven.
  - Ebbe a tömbbe válogassa ki az interfészt megvalósító elemeket.
  - Majd adja vissza ezt a tömböt visszatérési értékként.

OE.Prog2.Jatek.Keret névtéren belül készítsük el az alábbiakat:

A *Keret* osztályt egészítsük ki az alábbiakkal:

- Meglévő metódus módosítása: **Futtatas** – a metódus elején hozzunk létre egy második *KonzolosMegjelenito* objektumot is, ennek paraméterként átadva a Béla objektumot forrásként, és a 25,0 koordinátákat megjelenítési pozícióként.
- Meglévő metódus módosítása: **Futtatas** – a billentyűleütéseket figyelő ciklust egészítsük ki azzal, hogy minden iterációban meghívja ennek a megjelenítőnek is a *Megjelenites* metódusát.

Tesztelés: A fentieket megvalósítva működik a vezérlés és az üzleti logika, emellett elkészült a megjelenítési szint. Ezt rá tudjuk kapcsolni vagy az egész pályára, akkor mindent látunk (debug mód), vagy rá tudjuk kapcsolni az egyes játékosokra, ez lesz a végleges játék irányító képernyője.

4. Óra – Események (automatizmusok, ellenfelek)

OE.Prog2.Jatek.Szabalyok névtéren belül készítsük el az alábbiakat:

Új publikus osztály: **GepiJatekos**, ami a *Jatekos* osztály leszármazottja

A *GepiJatekos* osztályt egészítsük ki az alábbiakkal:

- Új konstruktor: **GepiJatekos** – Paraméterei ugyanazok mint az ős konstruktorának, csak hozzá továbbítsa a kapott adatokat.
- Új publikus metódus: **Mozgas** – Generáljunk egy véletlen számot 0 és 3 között. Majd ennek értékétől függően az objektum próbáljon meg elmozdulni (célszerű használni az örökölt *Megy* metódust) fel-jobbra-le-balra irányok közül valamerre.

- Új publikus felülírt mező: **Alak** – Visszatérési értéke legyen '\u2640'

Új publikus osztály: **GonoszGepiJatekos**, ami a *GepiJatekos* osztály leszármazottja

A *GonoszGepiJatekos* osztályt egészítsük ki az alábbiakkal:

- Új konstruktor: **GonoszGepiJatekos** – Paraméterei ugyanazok mint az ős konstruktorának, csak hozzá továbbítsa a kapott adatokat.
- Publikus felülírt mező: **Alak** – Visszatérési értéke legyen '\u2642'
- Új publikus metódus: **Utkozes** – Hívja meg az ős *Utkozes* metódusát. Ha még ezek után *Aktiv* a játékos, és a paraméterként átadott ütköző objektum egy *Jatekos* objektum (vagy annak leszármazottja), akkor hívja meg annak *Serul* metódusát paraméterként 10-et adva át.

OE.Prog2.Jatek.Keret névtéren belül készítsük el az alábbiakat:

A *Keret* osztályt egészítsük ki az alábbiakkal:

- Meglévő metódus módosítása: **Futtatas** – a metódus elején hozzunk létre egy *GepiJatekos* objektumot Kati néven, aki az eddigi játékossal egy térben egy másik ponton indul, illetve egy *GonoszGepiJatekos* objektumot Laci néven, aki az eddigi játékossal egy térben, de másik ponton indul (próbaképp létrehozhatunk megjelenítőket is ezekhez az objektumokhoz, hogy nyomon követhetjük a mozgásukat).
- Meglévő metódus módosítása: **Futtatas** – a billentyűleütéseket figyelő ciklust egészítsük ki azzal, hogy mindkét gépi játékosnak hívja meg a *Mozgas* nevű metódusát. Az ellenfelek ezzel mozogni kezdenek, de mindig csak akkor, amikor az emberi játékos is lép.

OE.Prog2.Jatek.Automatizmus névtéren belül készítsük el az alábbiakat:

Új publikus interfész: **IAutomatikusanMukodo**

Az *IAutomatikusanMukodo* interfészt egészítsük ki az alábbiakkal:

- Új metódus: **Mukodik** – ez a paraméter nélküli metódus automatikusan meg fog hívódni bizonyos időközönként.
- Új tulajdonság: **MukodesIntervallum** – csak olvasható egész (tizedmásodperc). Az interfészt megjelenítő objektum Mukodik metódusa ilyen időközönként fog meghívódni.

Töltsük le az **OrajelGenerator.cs** filet és a benne lévő **OrajelGenerator** osztályt másoljuk be ebbe a névtérbe. Ez az osztály egy időzítőt tartalmaz, amihez a *Felvetel* metódus segítségével fel tudnak iratkozni *IAutomatikusanMukodo* interfészt megvalósító objektumok. Az órajel-generátor *Aktivalas* metódusa tizedmásodpercenként lefut egyszer, és elvégzi az alábbi tevékenységeket: 1) Megnézi az összes feliratkozott objektumot, hogy most épp meg kell-e hívni a *Mukodik* metódust 2) Ha igen, akkor meghívja ezt a metódust.

OE.Prog2.Jatek.Szabalyok névtéren belül készítsük el az alábbiakat:

A *GepiJatekos* osztály valósítsa meg az *IAutomatikusanMukodo* interfészt az alábbiak szerint:

- Új publikus metódus: **Mukodik** – hívja meg a már létező *Mozgas* metódust.
- Új publikus tulajdonság: **MukodesIntervallum** – mindig adjon vissza 2-t (tehát 2 tizedmásodpercenként szeretnénk léptetni az ellenfelet).

OE.Prog2.Jatek.Keret névtéren belül készítsük el az alábbiakat:

A *Keret* osztályt egészítsük ki az alábbiakkal:

- Új privát mező: **generator** – egy *OrajelGenerator* típusú objektum legyen.

- Meglévő konstruktor módosítása: **Keret** - hozza létre az előző *generator* nevű objektumot.
- Meglévő metódus módosítása: **Futtatas** – a metódust módosítsuk úgy, hogy a létrehozott Kati és Laci objektumokat adjuk át paraméterként az *OrajelGenerator* objektum *Felvetel* metódusának. Az ellenfelek ezzel már maguktól mozognak, de a kép frissítése még mindig a billentyű lenyomásokhoz igazodik. Ha a fő ciklusban előzőleg közvetlenül hívtuk az ellenfelek *Mozgas* metódusát, akkor ezt már törölhetjük.

OE.Prog2.Jatek.Megjelenites névtéren belül készítsük el az alábbiakat:

A *KonzolosMegjelenito* osztály valósítsa meg az *IAutomatikusanMukodo* interfészt az alábbiak szerint:

- Új publikus metódus: **Mukodik** – hívja meg a már létező *Megjelenites* metódust.
- Új publikus tulajdonság: **MukodesIntervallum** – mindig adjon vissza 1-t (tehát minden órajel-ciklusban frissíteni kell a képernyőt).

OE.Prog2.Jatek.Keret névtéren belül készítsük el az alábbiakat:

A *Keret* osztályt egészítsük ki az alábbiakkal:

- Meglévő metódus módosítása: **Futtatas** – a metódust módosítsuk úgy, hogy a létrehozott megjelenítő objektumokat is kapcsoljuk rá az órajel-generátorra. Ezzel a képernyő frissítés is független a billentyűzet kezeléstől.

OE.Prog2.Jatek.Szabalyok névtéren belül készítsük el az alábbiakat:

Új delegált típus: **KicsFelvetelKezelo** – két paraméterrel rendelkezzen:

- A *Kincs* referenciája amit felvettek.
- A *Jatekos* referenciája, aki felvette.

A *Kincs* osztályt egészítsük ki az alábbiakkal:

- Új publikus esemény: **KincsFelvetel** – legyen egy *KincsFelvetelKezelo* típusú eseménykezelő.
- Meglévő metódus módosítása: **Utkozes** – amennyiben egy játékos felvette a kincset, és valaki feliratkozott a fenti eseménykezelőre, akkor küldjön az eseményről egy értesítést.

Új delegált típus: **JatekosValtozasKezelo** – három paraméterrel rendelkezzen:

- Az eseményt küldő *Jatekos* referenciája.
- A játékos új pontszáma.
- A játékos új életereje.

A *Jatekos* osztályt egészítsük ki az alábbiakkal:

- Új publikus esemény: **JatekosValtozas** – legyen egy *JatekosValtozasKezelo* típusú eseménykezelő.
- Meglévő metódus módosítása: **Serul** – amennyiben változott a játékos életereje, és valaki feliratkozott a fenti eseménykezelőre, akkor küldjön az eseményről egy értesítést.
- Meglévő metódus módosítása: **PontotSzerez** – amennyiben változott a játékos pontszáma, és valaki feliratkozott a fenti eseménykezelőre, akkor küldjön az eseményről egy értesítést.

OE.Prog2.Jatek.Megjelenites névtéren belül készítsük el az alábbiakat:

Új publikus osztály: **KonzolosEredmenyAblak**

A *KonzolosEredmenyAblak* osztályt egészítsük ki az alábbiakkal:

- Új privát mező: **pozX** – ennyivel kell majd eltolni a kirajzolandó adatok x koordinátáját (hasonló a *KonzolosMegjelenito*-höz).
- Új privát mező: **pozY** – ennyivel kell majd eltolni a kirajzolandó adatok y koordinátáját.
- Új privát mező: **maxSorSzam** – hány sor fér ki az eredményablakba.
- Új privát mező: **sor** – aktuális sorok száma (alapból 0).
- Új publikus konstruktor: **KonzolosEredmenyAblak** – adjon értéket az első három mezőnek.
- Új privát metódus: **JatekosValtozasTortent** – A *JatekosValtozasKezelo* delegáltnak megfelelő paraméterekkel rendelkezzen. A *SzalbiztosKonzol* segítségével írja ki a *pozX* és *pozY* által megadott helyre, az aktuális sornak megfelelő sorba az alábbi adatokat „játékos neve:…, pontszáma:…, életereje:…”. Majd növelje a *sor* értékét. Ha ez nagyobb mint a *maxSorSzam*, akkor állítsa 0-ra (ezzel elkezdi felülírni a legrégebbi eseményeket).
- Új publikus metódus: **JatekosFeliratkozas** – paraméterként egy *Jatekos* objektumot kap. Az objektum *JatekosValtozas* eseményére iratkozzon fel az előző metódussal.

OE.Prog2.Jatek.Keret névtéren belül készítsük el az alábbiakat:

A *Keret* osztályt egészítsük ki az alábbiakkal:

- Meglévő metódus módosítása: **Futtatas** – a metódust módosítsuk úgy, hogy hozzon létre egy új *KonzolosEredmenyAblak* objektumot a 0,12 pozícióban, maximálisan 5 sorral. Majd adjuk át a Laci játékos objektumot a *JatekosFeliratkozas* metódusának.
- Új privát mező: **megtalaltKincsek** – ez fogja számolni, hogy hány kincset vettek már fel.
- Új privát metódus: **KincsFelvetelTortent** – a *KincsFelvetelKezelo* delegáltnak megfelelő paraméterekkel rendelkezzen. Növelje a *megtalaltKincsek* változó értékét. Ha ez elérte a *KINCSEK_SZAMA* konstans értéket, akkor a *jatekVege* változo értéke legyen igaz.
- Meglévő metódus módosítása: **PalyaGeneralas** – a metódust módosítsuk úgy, hogy minden létrehozott *Kincs* objektum *KincsFelvetel* eseménykezelőjéhez kapcsoljuk hozzá az előző metódust.
- Új privát metódus: **JatekosValtozasTortent** – a *JatekosValtozasKezelo* delegáltnak megfelelő paraméterekkel rendelkezzen. Amennyiben a paraméterként kapott életerő 0, akkor állítsa a *jatekVege* változót igazra.
- Meglévő metódus módosítása: **Futtatas** – a metódust módosítsuk úgy, hogy az emberi játékos létrehozása után a *JatekosValtozas* eseményre iratkozzon fel az előző metódus.

Tesztelés: véletlenszerűen mozogniuk kell az ellenfeleknek, automatikusan frissülnie kell a képernyőnek, és a játékos eredményeinek meg kell jelennie az új ablakban. Mind a kincsek elfogyásakor, mind az életerő 0-ra csökkenésekor véget kell érnie a játéknak.

5. Óra – Kivételkezelés

OE.Prog2.Jatek.Szabalyok névtéren belül készítsük el az alábbiakat:

Új publikus osztály: **MozgasNemSikerultKivetel**, ami az *Exception* osztály leszármazottja

A *MozgasNemSikerultKivetel* osztályt egészítsük ki az alábbiakkal:

- Új privát mező: **jatekElem** – ez tárolja, hogy ki nem tudott lépni.
- Új publikus tulajdonság: **JatekElem** – a fentit adja vissza.
- Új privát mező: **x** – ez tárolja, hogy hova szerett volna lépni (x koordináta)
- Új publikus tulajdonság: **X** – a fentit adja vissza.
- Új privát mező: **y** – ez tárolja, hogy hova szerett volna lépni (y koordináta)
- Új publikus tulajdonság: **Y** – a fentit adja vissza.
- Új publikus konstruktor: **MozgasNemSikerultKivetel** – beállítja az előzőket.

Új publikus osztály: **MozgasHalalMiattNemSikerultKivetel,** ami a *MozgasNemSikerultKivetel* osztály leszármazottja

A *MozgasHalalMiattNemSikerultKivetel* osztályt egészítsük ki az alábbiakkal:

- Új publikus konstruktor: **MozgasHalalMiattNemSikerultKivetel** – a szükséges adatokat továbbítja az ősének.

Új publikus osztály: **MozgasHelyHianyMiattNemSikerultKivetel,** ami a *MozgasNemSikerultKivetel* osztály leszármazottja

A *MozgasHelyHianyMiattNemSikerultKivetel* osztályt egészítsük ki az alábbiakkal:

- Új publikus konstruktor: **MozgasHalalMiattNemSikerultKivetel** – a szükséges adatokat továbbítja az ősének.

-

A *MozgasNemSikerultKivetel* osztály valósítsa meg az *IAutomatikusanMukodo* interfészt az alábbiak szerint:

- Új publikus metódus: **Mukodik** – hívja meg a már létező *Mozgas* metódust.

Új publikus tulajdonság: **MukodesIntervallum** – mindig adjon vissza 2-t (tehát 2 tizedmásodpercenként

6. Óra – Backtrack (labirintus készítés, kincsek elhelyezése)

7. Óra – Láncolt lista (memória megvalósítása)

8. Óra – Bináris keresőfa (?)

9. Óra – Gráf (ellenség okosítása, útkereséssel)