I. forduló (Hagyományos verseny) 2025. november 7-9.

Damareen – a gyűjtögetős fantasy kártyajáték, amelyben stratégia, szerencse és képzelet fonódik össze. A selyemutak játszóasztalaitól a modern digitális arénákig ez a műfaj mindig is a hősök és történetek kovácsa volt. Most rajtad a sor, hogy saját paklid lapjaira írd a történelmet: hősöket teremts, kazamatákon küzdj végig, és szörnyek vezéreivel mérkőzz meg. Vajon a gondosan kidolgozott stratégiád diadalt arat, vagy a kazamaták mélye örökre elnyel? Készítsd elő a paklidat, mert a kártyák sorsot hordoznak!

A csapatotok feladata, hogy megvalósítsa a **Damareen** játékprogramot, amely a jelen leírásban szereplő szabályok szerint működik.

A programot úgy kell megírni, hogy kétféle üzemmódban is működjön:

- **teszt mód:** az alkalmazás a paraméterként megadott **abszolút elérésű** könyvtárban elhelyezett *in.txt* bemeneti fájlon keresztül vezérelhető. Az eredményeket a mappán belül az *in.txt*-ben megadott kimeneti fájlokba írja (ld. még később). Ez a mód az alkalmazás alapműködése.
- **játék mód**: ez biztosítja, hogy a játék a felhasználói felületről is játszható legyen. Ez a *--ui* parancssori argumentummal aktivizálható. Ilyenkor az alkalmazás konzolos vagy grafikus felületen keresztül játszhatóvá válik.

Tehát ha a lefordított program fájlneve *cardgame.exe*, akkor a:

```
cardgame.exe c:\tesztek\teszteset_01\
```
→ teszt módban indítja a programot, az *in.txt* állomány a *c:\tesztek\teszteset_01* mappában található, és ide kerülnek mentésre a kimeneti fájlok is.

```
cardgame.exe --ui
```
→ játék módban indítja a programot.

Természetesen interpretált nyelv esetén (pl. Python) vagy virtuális gép által futtatott nyelv esetén (Java) nincs lefordított *.exe* állomány, de paraméterként ugyanúgy fogadni kell tudni a bemeneti mappa nevét vagy a *--ui* kapcsolót.

1. A játékmenet teszt módban

A játék célja, hogy a játékos a saját kártya gyűjteményét az ún. kazamaták ellen vívott harcok során fejlessze. Minél erősebbek lesznek a játékos kártyái, annál komolyabb ellenfeleket tud legyőzni, és ezzel a nyeremények nagysága is nő.

A játéknak soha nincs vége, a kártyák a végtelenségig fejleszthetőek.

A játékot legkönnyebben egy példa játékmeneten keresztül lehet jól megérteni, amit az *in.txt* állományon keresztül vezérlünk (max. 200 soros). Ezen az állományon keresztül vezérelhető a játék, és lekérdezhető annak állapota output fájlokba történő írással, így tesztelhető automata módon a játék logikájának helyes megvalósítása.

Feltételezhetitek, hogy az *in.txt* állomány formátuma és az abban szereplő input adatok az alábbi leírásnak megfelelőek, és csak ASCII karaktereket tartalmaznak.

Az *in.txt* állomány bármikor tartalmazhat egy vagy több egymás utáni üres sort.

1.1 A világ létrehozása

Mielőtt a játék elkezdődik, a játékmester létrehozza a **világot**. A világ a következő részekből áll:

- világkártyák: ezek lehetnek ún. sima kártyák vagy vezérkártyák.
- kazamaták: ezek a játék kihívás pályái, világkártyák sorozatából állnak.

Az ábrán egy világhoz tartozó sima kártyákat láthatunk. A kártyák sorrendje számít, ezek így egy sorozatot alkotnak. Az első kártya jelentése a következő (a többi kártya is e mentén a logika mentén értelmezhető):

- ***Aragorn*** a kártya neve (egyedi a világkártyák között), Max. 16 karakter hosszú
- ***2*** a kártya **sebzés értéke**: a kártya alapesetben ekkora erővel "üti meg" az ellenfél kártyát egy ütközet során. Legalább 2, egész. Max. 100 lehet. (Nem lesz tesztelve nagyobb értékre.)
- ***5*** a kártya **életereje**: Minden harc kezdetekor erről az életerőről indul a kártya, ami az ellenfél sebzései során az ellenfél sebzési értékével csökken. Legalább 1, egész. Max. 100 lehet. (Nem lesz tesztelve nagyobb értékre.)
- ***tűz*** a kártya típusa. Négy típus létezik: föld, levegő, víz, tűz. A harcban álló lapok típusai befolyásolják a támadó kártya aktuális sebzés értékét és ezáltal a védekező kártya életerejének csökkenését (ld. még később).

Az *in.txt* állomány mindig a sima kártyák meghatározásával kezdődik, és így néz ki a fenti esetben:

`in.txt:`

```
uj kartya;Aragorn;2;5;tuz
uj kartya;Sadan;2;4;levego
uj kartya;Corky;2;4;fold
uj kartya;Kira;2;7;levego
uj kartya;Eowyn;2;5;viz
uj kartya;ObiWan;2;2;fold
uj kartya;Tul'Arak;2;4;fold
```

A típus érték *fold*, *levego*, *viz* vagy *tuz* lehet.

A világ részei az ún. **vezérkártyák** is, amiket a játékmester a sima kártyákból származtat. Ezek önálló kártyaként, saját elnevezéssel bírnak. A vezérkártya örökli az eredeti sima kártya típusát, de a származtatás típusa szerint módosul sebzés értékben **vagy** életerőben:

- **sebzés duplázás** esetén a sebzés érték kétszereződik,
- **életerő duplázás** esetén az életerő kétszereződik.

A példában az *ObiWan* kártyából (sebzés: 2, életerő: 2) sebzés duplázással létrehozzuk a *Darth ObiWan* kártyát (sebzés: 4, életerő: 2).

Az *in.txt* állományban a vezérkártyákat a sima kártyák után határozzuk meg ebben a formátumban (fenti ábrához):

`in.txt:`

```
uj vezer;Darth ObiWan;ObiWan;sebzes
```

vagy (életerő duplázás esetén):

`in.txt:`

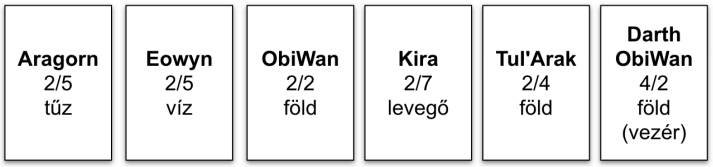
```
uj vezer;Darth ObiWan;ObiWan;eletero
```

A vezérkártya neve max. 16 karakterből állhat.

A világhoz tartoznak a kazamaták is. A kazamatával történő egy-egy harc során a játékos sima kártyák és vezérek sorozatával mérkőzik meg jutalomért cserébe.

A kazamatáknak három típusa létezik:

Típus	Kártyák (a legyőzendő ellenség lapok száma)	Nyeremény a kazamata legyőzése esetén
Egyszerű találkozás	1 sima	A játékos által utoljára kijátszott kártya (ld. később): ● vagy: +1 sebzést kap ● vagy: +2 életerőt kap A kazamatának a paramétere, hogy plusz sebzést vagy plusz életerőt lehet kapni a kazamata legyőzéséért.
Kis kazamata	3 sima+1 vezér	
Nagy kazamata	5 sima+1 vezér	1 **sima** kártyát kap a játékos a gyűjteményébe a világból - az első olyat a világból, ami még nincs a gyűjteményében.



Az ábrán szereplő, *Teszt3 Kazamata* elnevezésű nagy kazamata így írható le:

`in.txt:`

```
uj kazamata;nagy;Teszt3 Kazamata;Aragorn,Eowyn,ObiWan,Kira,Tul'Arak;Darth
ObiWan
```

A kazamatáknak összesen 5 variációja lehetséges az alábbi példák szerint:

`in.txt:`

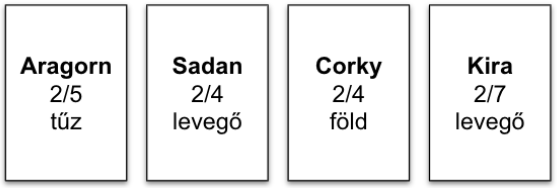
```
uj kazamata;egyszeru;Teszt1a Kazamata;Sadan;eletero
uj kazamata;egyszeru;Teszt1b Kazamata;Aragorn;sebzes
uj kazamata;kis;Teszt2a Kazamata;Aragorn,Eowyn,ObiWan;Darth ObiWan;eletero
uj kazamata;kis;Teszt2b Kazamata;Sadan,Eowyn,Kira;Darth ObiWan;sebzes
uj kazamata;nagy;Teszt3 Kazamata;Aragorn,Eowyn,ObiWan,Kira,Tul'Arak;Darth
ObiWan
```

Az *in.txt* fájlban egy kazamata leírásánál először tehát az új kazamata létrehozás vezérlő kifejezése szerepel (*uj kazamata*), majd a típus (*egyszeru*, *kis*, *nagy*), utána a kazamata egyedi neve (pl. *Teszt1a Kazamata*), majd a sima kártyák vesszővel felsorolva, aztán a vezér kártya (kis és nagy kazamata esetén), majd a nyeremény típusa (egyszerű találkozás vagy kis kazamata esetén: *sebzes* vagy *eletero*, nagy kazamata esetén kihagyva). Ugyanaz a sima kártya csak egyszer használható fel ugyanabban a kazamatában, de egy másik kazamatában már újra szerepelhet. A kazamata neve max. 20 karakter hosszú lehet.

1.2 A világkártyák és azokból játékos gyűjtemény összeállítása

Az eddigiek során a játékmester a világot definiálta, a következő lépésben pedig folytatja a játékos ún. **gyűjteményének** létrehozásával. A gyűjtemény a játékos saját kártyáinak teljes készlete, amit a világkártyák közül egyes **sima kártyák** felvételével tesz meg. A gyűjteményben szereplő kártyák lesznek a játékos saját kártyái, amiket a játék harcai során fejleszt.

Az alábbi példában a játékmester összeállította a fenti világ sima kártyáiból a játékos gyűjteményét:



`in.txt`:

```
uj jatekos

felvetel gyujtemenybe;Aragorn
felvetel gyujtemenybe;Sadan
felvetel gyujtemenybe;Corky
felvetel gyujtemenybe;Kira
```

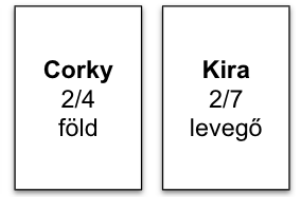
Ugyanaz a kártya csak egyszer szerepelhet a játékos gyűjteményében, de szerepelhet bármely kazamatában is.

Az *uj jatekos* sor (pontosan egyszer szerepel egy *in.txt* fájlban) választja el a világkártya és kazamata meghatározó részt a gyűjteménybe felvételtől.

1.3 Pakli összeállítása

Most következhet végre maga a játék rész, azaz a harcok sorozata! Ezt már a játékos irányítja.

A játékos csak úgy indulhat harcba, ha már van ún. **paklija** összeállítva. A pakli a gyűjtemény legfeljebb felét kitevő, rögzített sorrendű lapok csoportja, amelyet a játékos harc előtt állít össze. (Ha a gyűjteményben páratlan számú lap szerepel, akkor felfelé kell kerekíteni.) A pakliban lévő kártyák sorrendje különbözhet attól a sorrendtől, ahogy ezek a kártyák egymáshoz képest a gyűjteményben szerepelnek. A játékosnak mindig (legfeljebb) egy paklija van, de a játék során akárhányszor újra definiálhatja azt.



in.txt:

```
uj pakli;Corky,Kira
```

Az *uj pakli* mindig csak a gyűjtemény meghatározása után szerepelhet, és akárhány alkalommal ismétlődhet.

1.4 A harc

A **harcot** az alábbi parancs indítja:

in.txt:

```
harc;Teszt1a Kazamata;out.harc01.txt
```

A *harc* a harchoz tartozó vezérlő utasítás, a *Teszt1a Kazamata* a kihívást jelentő kazamata neve, az *out.harc01.txt* állomány pedig az a fájl, ahová a harc ütközeteit szükséges naplózni.

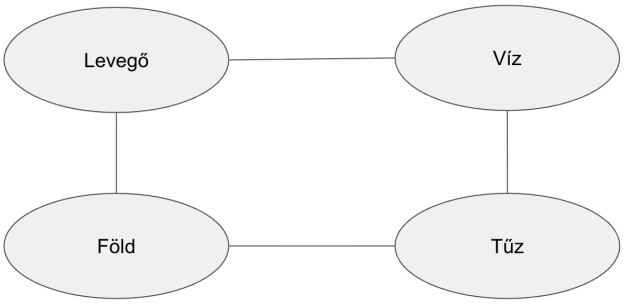
Harc kezdetekor a két ellenfél serege felsorakozik: Az egyik oldalon sorban a játékos paklijának lapjai, másik oldalon pedig a kiválasztott kazamata kártyái sorakoznak fel, ez utóbbi a játékmester által felvitt sorrendben. Minden kártya életereje "alaphelyzetbe kerül".

Az első ütközetben a játékos paklijának első kártyája és a kazamata első kártyája ütközik össze. Először a kazamata játssza ki a lapját, majd a játékos. A kazamata üt, és sebzés értékével csökkenti a védekező - játékos - kártyájának életerejét. Ha a játékos kártyája túléli (életereje nagyobb mint 0), akkor ő lesz a támadó, míg a kazamata a védekező.

Ha elesik a játékos kártyája (életereje 0 vagy kevesebb lesz - ez utóbbi esetben is 0-val kell számolni az életerőt), akkor az az adott ütközetben már nem játszik tovább (kiesik), és helyét a sorrendben következő kártya veszi át a pakliból - ha még van a pakliban kártya. A kazamata újra támad.

Ha a játékos ütése esetén a kazamata aktuális kártyája elesik, akkor a kazamata a következő lapját játssza ki, és folytatódik a kör ugyanazon szabályok szerint.

A támadó kártya sebzés értéket befolyásolhatja típusának és a védekező kártya típusának viszonya az alábbiak szerint:



- Amennyiben a két kártya típusa egymásra "**erős**" (az ábrán szomszédos csúcsok), akkor a támadó lap **kétszeres sebzés értékkel** üti meg a védekező kártyát. Pl. ha a támadó "levegő" típusú, a védekező "föld", akkor ezek egymásra erősek, így a támadó kártya kétszeres sebzés értékkel üti meg majd a másikat.
- Amennyiben a két kártya típusa egymásra "**gyenge**" (az ábrán ellentétes sarokban lévő csúcsok), akkor a támadó lap **1/2 sebzés értékkel** üti meg a védekező kártyát. Pl. ha a támadó "levegő" típusú, a védekező "tűz", akkor ezek egymásra gyengék, így a támadó lap 1/2 sebzés értékkel üti meg majd a másikat. Páratlan sebzés érték esetén a felezett sebzésértéket lefelé kell kerekíteni.
- Amennyiben a két kártya típusa se nem erős, se nem gyenge (azaz ugyanaz a típusuk), akkor nem módosul a támadó eredeti sebzés értéke.

A harc akkor ér véget, amikor valamelyik fél összes kártyája elesik. Ha a harc során a játékos nyer, akkor a kazamata típusától függően a korábbiakban említettek szerint gyűjteménye fejlődik. Az ütközetek csak a harc kimenetelét befolyásolják, a játékos gyűjteményéből egyetlen lap sem kerül ki véglegesen. A gyűjtemény kártyái kizárólag nyeremény hatására változnak (fejlődés, új lap).

A harc menete az alábbi egyszerű példán keresztül jól megérthető.

A harc követhetőségét **körökre** osztva, **akciók** sorozataként (**kijátszás** vagy **támadás**) lehet a legjobban elősegíteni. Egy körhöz pontosan egy-egy, a kazamatához és a játékoshoz tartozó akció tartozik. Az alábbi ábrán az 1/1 az első kör első akcióját, az 1/2 az első kör második akcióját jelenti (stb. így tovább), és ezek hatását mutatja.


A játékos paklija
A játékos és a kazamata kijátszott lapja
A kazamata kártyái
Corky 2/4 föld
Kira 2/7 levegő
Sadan 2/4 levegő
Kira 2/7 levegő
Corky 2/4 föld
Sadan 2/4 levegő
1/2
1/1
A kazamata majd a játékos is kijátssza a soron következő lapját.
Kira 2/7 levegő
Corky 2/0 föld
Sadan 2/4 levegő
2/1
A kazamata üt. Mivel a levegő és föld erősek egymásra, 4-es ütés után Corky életereje 0-ra csökken.
Kira 2/7 levegő
Sadan 2/4 levegő
2/2
A játékos kijátssza a következő lapját.
Kira 2/5 levegő
Sadan 2/4 levegő
3/1
A kazamata újra támad 2-es sebzéssel (levegő és levegő se nem erős, se nem gyenge egymásra.)
Kira 2/5 levegő
Sadan 2/2 levegő
3/2
A játékos támad, 2-es sebzéssel üt.
Kira 2/3 levegő
Sadan 2/2 levegő
4/1
Újra a kazamata támad, 2-es sebzéssel üt.
Kira 2/3 levegő
Sadan 2/0 levegő
4/2
A játékos támad, a kazamata lapjának életereje 0 lesz. A játékos nyer.

Ha az utolsó körben a kazamata nyer, akkor abban a körben csak egyetlen akció születik, oda már nem kerül játékoshoz tartozó akció.

A harc menetének naplózását a következőképpen szükséges elvégezni a megadott kimeneti állományba:

`out.harc01.txt:`

```
harc kezdodik;Teszt1a Kazamata

1.kor;kazamata;kijatszik;Sadan;2;4;levego
1.kor;jatekos;kijatszik;Corky;2;4;fold

2.kor;kazamata;tamad;Sadan;4;Corky;0
2.kor;jatekos;kijatszik;Kira;2;7;levego

3.kor;kazamata;tamad;Sadan;2;Kira;5
3.kor;jatekos;tamad;Kira;2;Sadan;2

4.kor;kazamata;tamad;Sadan;2;Kira;3
4.kor;jatekos;tamad;Kira;2;Sadan;0

jatekos nyert;eletero;Kira
```

A játékos nyert, és nyeréskor a nyereményt is naplózni szükséges. Jelen esetben az *Teszt1a kazamata* nevű egyszerű kazamata ellen játszott, aminek nyereménye +2 életerő (létrehozásakor definiált), ezért a játékos utolsó támadásban résztvevő lapja *(Kira)* kap +2 életerőt, így a következő harc során már 9 életerőről indul ez a kártya, ez lesz az (új) alaphelyzete. Ha több harcban is nyer a kártya, akkor "halmozódik" a nyeremény, így a kártya tetszőlegesen sokáig fejleszthető.

Az utolsó sor formátuma a következő lehet:

Utolsó sor (példa)	**Eset**
`jatekos nyert;eletero;ObiWan`	A játékos nyert. A kazamata típusa egyszerű találkozás vagy kis kazamata, a nyeremény típusa **életerő**. ObiWan a kártya neve, aki fejlődik, és így +2 életerőt szerez.
`jatekos nyert;sebzes;ObiWan`	A játékos nyert. A kazamata típusa egyszerű találkozás vagy kis kazamata, a nyeremény típusa **sebzés**. ObiWan a kártya neve, aki fejlődik, és így +1 sebzést szerez.
`jatekos nyert;Eowyn`	A játékos nyert. A kazamata típusa nagy kazamata. A játékos nyereménye ezért az, hogy a világból az **első olyan sima kártyát** (példában Eowyn) megkapja, ami még nem szerepel a gyűjteményében.
`jatekos vesztett`	A **játékos vesztett**.

Ha egy harc befejeződött, akkor vagy a már összeállított paklival, vagy egy új pakli összeállításával kezdhető új harc egy tetszőleges kazamata ellen (ami lehet az utolsó kazamata vagy egy másik is). Új pakli összeállítása esetén az előző pakli "érvényét veszti".

1.5 Játékállapot (export funkciók)

Teszt módban lehetőség van a játék állapotának szoros nyomkövetésére a világ és a játékos állapotának exportjával.

in.txt:

```
export vilag;out.vilag01.txt
```

out.vilag01.txt:

```
kartya;Aragorn;2;5;tuz
kartya;Sadan;2;4;levego
kartya;Corky;2;4;fold
kartya;Kira;2;7;levego
kartya;Eowyn;2;5;viz
kartya;ObiWan;2;2;fold
kartya;Tul'Arak;2;4;fold

vezer;Darth ObiWan;4;2;fold

kazamata;egyszeru;Teszt1a Kazamata;Sadan;eletero
kazamata;egyszeru;Teszt1b Kazamata;Aragorn;sebzes
kazamata;kis;Teszt2a Kazamata;Aragorn,Eowyn,ObiWan;Darth ObiWan;eletero
kazamata;kis;Teszt2b Kazamata;Sadan,Eowyn,Kira;Darth ObiWan;sebzes
kazamata;nagy;Teszt3 Kazamata;Aragorn,Eowyn,ObiWan,Kira,Tul'Arak;Darth ObiWan
```

in.txt:

```
export jatekos;out.jatekos02.txt
```

out.jatekos2.txt **(a fenti harc után):**

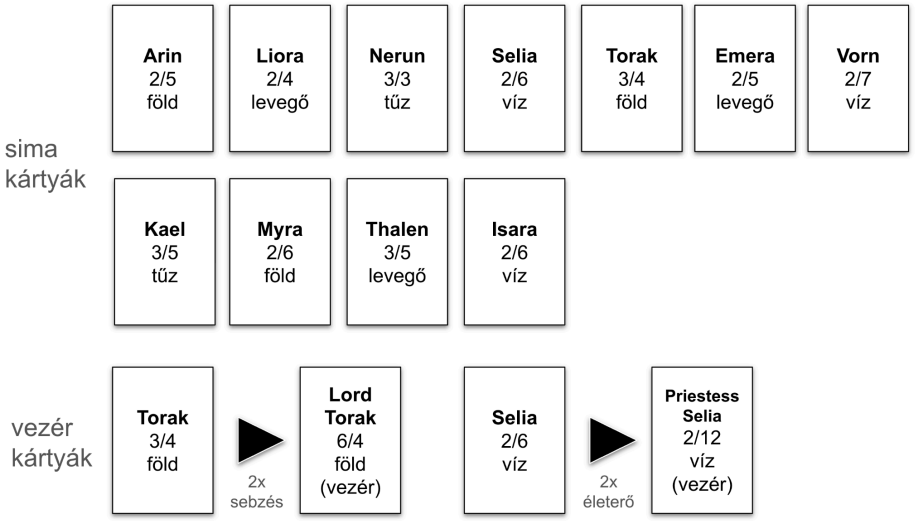
```
gyujtemeny;Aragorn;2;5;tuz
gyujtemeny;Sadan;2;4;levego
gyujtemeny;Corky;2;4;fold
gyujtemeny;Kira;2;9;levego

pakli;Corky
pakli;Kira
```

A verseny GitHub repositoryjában előre elkészített teszteseteket találhattok, amik segíthetnek a feladat megértésében és az alkalmazásotok tesztelésében. Az *in.txt*-ket és az egyes *in.txt*-k alapján az elvárt kimeneti fájlokat tesztesetenként külön mappában találhatjátok a *tesztesetek* mappában.

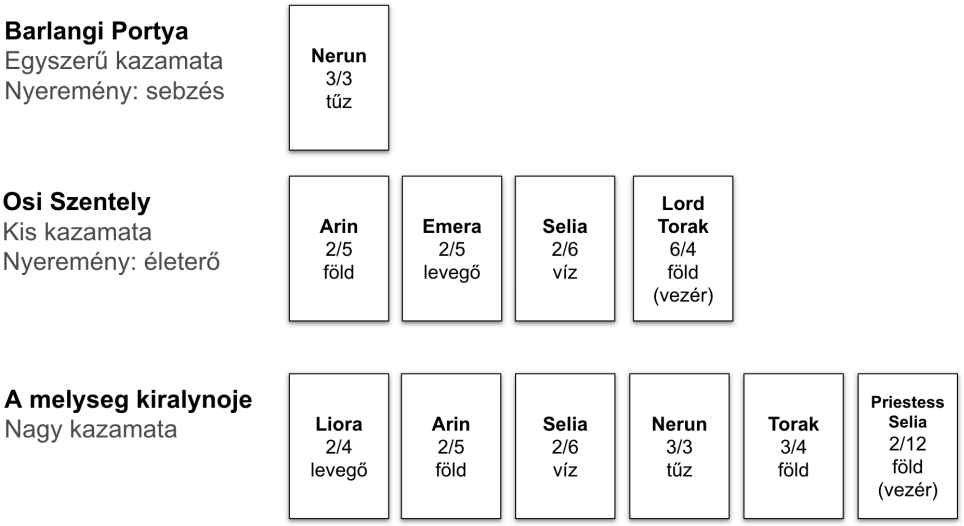
2. Játék mód

A játék módban előre definiált világkártyákkal, kazamatákkal és játékos gyűjteménnyel lehet játszani játékosként. (A játékmester szerepkört nem kell megvalósítani.)

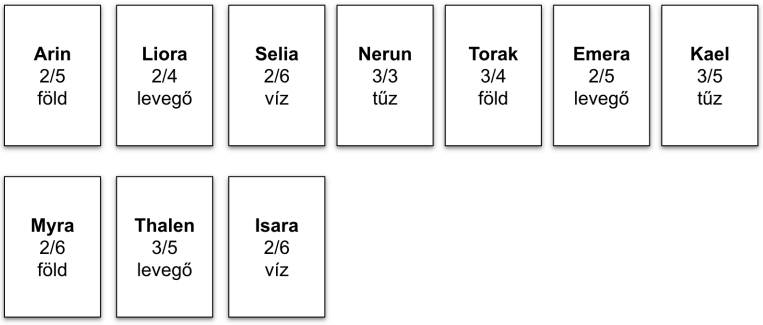
A világkártyák a következők (a sima kártyák a felvitel sorrendjében szerepelnek balról jobbra, két sorba tördelve):



A világban 3 kazamata létezik az alábbiak szerint (kártyák sorrendje balról jobbra):



A játékos gyűjteménye pedig az alábbi (a kártyák a felvitel sorrendjében balról jobbra szerepelnek két sorba tördelve):



A program indításakor egy új játék kezdődik. A program megjeleníti a világkártyákat, a kazamatákat és a játékos gyűjteményét.

Harc indításához kell lennie összeállított paklinak, és kazamatát kell kiválasztania a játékosnak.

Nagy kazamata ellen csak akkor lehet harcot indítani, ha a világban van olyan sima kártya, amivel a játékos még nem rendelkezik a gyűjteményében (tehát még nyerhet valamit). Teszt módban feltételezhetitek, hogy ilyen nem fordul elő, de játékos módban ezt az esetet is kezelni szükséges a programnak.

Harc közben nyomon követhetőnek kell lennie, hogy a harcban hogyan alakultak az egyes körök és azokban az akciók, az életerők hogyan változtak, hogyan változott a pakli, és hogy a végén ki nyert. Nem szükséges az egyes lépések között "léptetni", az is megfelelő, ha a harc lépései egyszerre kerülnek kiírásra.

Ha a játékos nyer, akkor kerüljön megjelenítésre a nyereménye.

Minden harc végén dönthet úgy a játékos, hogy új paklit állít össze. Új harc kezdetekor pedig akár másik kazamatát választhat.

Ügyeljetek arra, hogy a játék használata során hibás felhasználói input esetén az alkalmazás "ne omoljon össze". Adjon hasznos hibajelzést a program, és kényelmesen folytatható legyen a működés.

Legyen egyértelmű a felhasználó számára, hogy a programot hogyan kell használni, ehhez világos utasításokat adjatok azon felhasználók számára is, akik nem ismerik még a programotokat. (Azt feltételezhetitek, hogy a felhasználó ismeri magát a játékot és annak szabályait, így ezeket nem szükséges elmagyaráznotok.)

Folyamatosan és egyértelműen követhető legyen, hogy az elkezdett játék során milyen kártyák tartoznak a játékos gyűjteményébe és mik a paklijába.

3. Beadandó

Az alkalmazásotok **forrásfájljait** a csapatotok számára biztosított távoli Windows Server számítógépre kell elhelyeznetek a *c:\verseny* mappába.

A távoli szerverre előre telepített IDE-ken ill. fordítóprogramokon keresztül kérjük az alkalmazás fordítását és/vagy függőségeinek letöltését. Amennyiben nem áll rendelkezésetekre a kívánt fordítóprogram vagy build környezet, akkor azt feltelepíthetitek a gépre (admin jogosultságot biztosítunk). Ügyeljetek arra, hogy a Windows Server gép a verseny végéig működőképes maradjon, nincs lehetőségünk biztosítani tiszta ("szűz") gépet, amennyiben a csapat által telepített alkalmazások miatt a biztosított Windows Server használhatatlanná válna!

A következő IDE-ket találhatjátok a Windows Server gépen:

- Visual Studio Code 1.90 a következő kiegészítőkkel:
  - Live Server (Ritwick Dey) 5.7 vagy újabb
  - Python (Microsoft) 2023.14 vagy újabb
  - Pylance (Microsoft) 2023.8 vagy újabb
- PyCharm Community 2024.1 with Python 3.12
- Code::Blocks 20.03 MinGW/GCC
- MS Visual Studio Community 2022 (desktop development Visual Basic, Visual C#, Visual C++)

Ha a csapat Java-t használ, akkor a szükséges jdk-t, build toolt és szükséges egyéb szoftverkomponenseket telepítenie kell a rendelkezésre álló számítógépre.

A *c:\verseny* mappába kérjük elhelyezni a *run.bat* nevű állományt. Ennek feladata, hogy meghívja az általatok lefordított *.exe* állományt vagy - interpretált nyelv esetén (pl. Python) - meghívja a futtató programot. A *run.bat* fájlnak átadott paramétert tovább kell adni a programotoknak, ezáltal biztosítva az alkalmazás teszt és játékos módban történő elindításának lehetőségét.

Példa *run.bat* állományra, amennyiben az alkalmazás Pythonban íródott:

`run.bat` (Pythonhoz):

```
python main.py %1
```

Példa *run.bat* állományra, amennyiben az alkalmazás C#-ban íródott, és konzol alkalmazás:

`run.bat` (C#-hoz):

```
cd c:\verseny\bin\Debug\net7.0
kartya.exe %1
```

Ezek a *run.bat* állományok csak példák, a *run.bat*-ot úgy írjátok meg, hogy az általatok létrehozott programot indítsa el!

4. Értékelési szempontok

Az elkészített alkalmazás játék módban konzolos alkalmazás vagy grafikus felülettel rendelkező (Graphical User Interface-es, azaz GUI-s) alkalmazás is lehet. **Pontozás szempontjából egyik sem jelent előnyt a másikhoz képest a verseny 1. fordulójában.**

Teszt módban "tiszta" konzolos alkalmazásként kell működnie az alkalmazásnak, azaz ne várjon felhasználói interakcióra, és futás után automatikusan záródjon be. Futás közben tilos távoli szerverhez kapcsolódni!

Ha a *run.bat* állományt nem vagy rosszul hozzátok létre, akkor az alkalmazásotok az értékelés során 0 pontot fog kapni!

A versenybizottság teszt módban és játék módban is fogja ellenőrizni az alkalmazást. A forráskód nem kerül majd se elemezésre, se módosításra, így nagyon fontos, hogy futtatható és értékelhető megoldásokat készítsetek elő a Windows Server számítógépre!

A teszt mód használatának segítségével kerül teljeskörűen tesztelésre a játék logikájának helyes megvalósítása. Ennek kiértékelése automata módon fog történni, kisméretű tesztállományok segítségével, így nem szükséges arra felkészülnötök, hogy nagyméretű inputot hatékonyan kezeljetek.

Az alkalmazásotok manuális tesztelésen is át fog esni, ehhez az alkalmazásotok játék módban kerül majd futtatásra.

Annak érdekében, hogy teszt módban és játék módban is ugyanúgy működjön a játék logikája, a versenybizottság azt javasolja, hogy a játék logikája a kódban egyszer kerüljön megvalósításra, és ezt használja fel mind a teszt mód, mind a játék mód!

A fejlesztés során használhattok mesterséges intelligenciát (AI-t). A feladat szövegének megértéséhez, kérdések feltételéhez és saját tesztesetek elkészítéséhez a versenybizottság kifejezetten javasolja ennek használatát! A programozást tekintve viszont azt javasoljuk, hogy a generált programkód tüzetes ellenőrzésével és/vagy előre definiált teszteseteken keresztül ellenőrizzétek a programkódot, így garantálható a szoftver helyessége és az értékelésnél a magas pontszám!

Pontozandó elem	**Maximális pontérték**
Helyes működés (tesztmód futtatásával)	200 pont
Játszhatóság (játék mód futtatásával)	100 pont
Összesen	**300 pont**

A 2. fordulóban a továbbjutó csapatok az 1. fordulóban készített alkalmazást fejleszthetik tovább (akár újra implementálással is).

Jó munkát kíván a versenybizottság!