**Atestat a competenţelor profesionale  
Clasa matematică – informatică**

**Dömös Mózes  
12.A**

**Társasjáték:  
Snapszer**

**Szakirányítók:  
Szélyes Emőke  
Godra Hajnal**

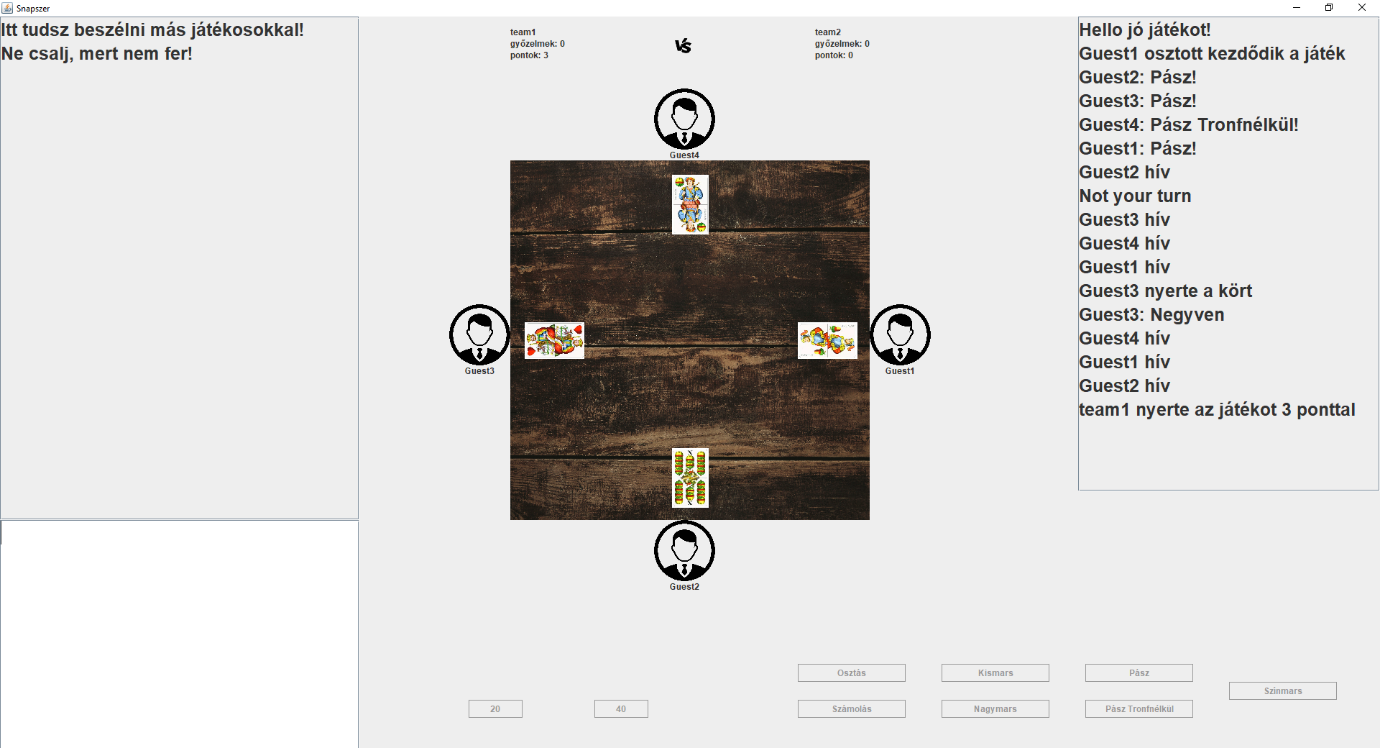
**-2022-**

**Tartalomjegyzék**

* **Projekt tulajdonságai 3**
* **Kinézet 3**
* **Játékmenet 4**
* **Project összetétele 6**
* **Néhány érdekes metódus 10**
* **Projekt Működése, interakciók, játék lefolyása 12**
* **Fejlesztési lehetőségek 15**
* **Végszó 16**

**Projekt tulajdonságai:**

* **Nyelv: Java (12.0.2 verzió)**
* **Fejlesztői környezet: IntelliJ IDEA**
* **Grafikus felhasználói felület (GUI): Java Swing**
* **Kommunikációs típus: Kliens-szerver modell**
* **Játék típusa: Többjátékos bonyolult kártyajáték**
* **Összetétele: 14 class**

**Kinézet**:

**Játékmenet**

A snapszer, más néven snapszli vagy hatvanhat népszerű, viszonylag bonyolult kártyajáték. Magyar kártyával vagy francia kártyával játszhatja 2-4 játékos. A játék célja legalább 66 pont elérése fogásokból, vagy a marsok bemondása. A megírt verzió a snapszer 4 játékossal való játékot teszi lehetővé. Ilyenkor 2 csapat van, általában a szemben üllő játékosok egy csapatban vannak.  
A meccs 21 játszmapontig megy, különféle játékmódokra különböző pontok járnak.  
Ha kigyűl a 21 játszmapont akkor az a csapat nyert 1 meccset, akármeddig lehet folytatni nincs megadva hány meccset kell nyerni, de általában BO3-at (best of 3) szoktak játszani.

A játék osztásokból áll, a játkosok egymás után osztanak, az óra járásával megegyező irányban, egy osztás során az osztó mindenkinek oszt 5 lapot, és az osztó 3-ik lapja a tronf. Az osztás után mindenkinek van lehetősége bemondani valamit sorba az osztó utáni játékostól kezdve, az óra mutatójával megegyező irányban.  
5 féle bemondás van: pász, pász tronfnélkül, kismars, nagymars, színmars.  
A marsoknak növekvő sorrendben vannak a fenti példában, ennek az a lényege, hogy csak egyszerre 1 mars lehet érvényben, pl. ha mondtak egy kicsimarsót és azután egy nagymarsót akkor a nagymars van érvényben, ha több azonos rangú marsot mondtak, akkor annak a játékosnak a marsa van érvényben aki legelőször mondta.  
A bemondásoktól függ a jelenlegi játék menete, 5 különböző játékmód van:

1.Újra osztás: ez akkor van érvényben mikor, az osztó elleni csapatnak nincs tronfja (egyik tagjának sem), pl ha az osztó az 1-es játékos akkor ha a 2-es és 4-as játékosnak nincs tronfja és bemondják azt akkor bedobják a lapokat és az osztó újra osztja őket.

2.Normális menet: ez akkor van ha az ellenséges csapatból legalább 1 valakinek van tronfja, vagyis akkor nem tudják bedobni a lapokat, és senki se mondott marsot, mindenki pászolt.

Ez a játék mód arról szól, hogy az osztó utáni játékos hív egy lapot, arra sorban (úgy mint a bemondásoknál) kell tegyenek egy ugyan olyan színű lapot, ha nincs akkor tronfot, ha az sincs akkor mindegy.   
  
Miután van 4 lap az asztalon aki a legnagyobbat rakta bele az viszi, kivéve ha volt benne tronf, akkor a legnagyobb tronf viszi. A következő lapot az a játékos hívja aki vitte a jelenlegi kört. Akkor van vége a játéknak, ha valakinek kigyűl a 66 pont, a vitelben lévő lapok összege az a fogópontszám ez a jelenlegi játék alatt megmarad és ehhez kell még gyűjteni. Mikor megvan a 66, ha az ellenséges csapatnak nincs fogása, akkor 3 játszmapont kaptok, ha van de nincs meg benne a 33 pont, akkor 2 játszmapontot, ha megvan a 33 pont akkor 1 játszmapontot.  
A játék közben aki hív az megtudja számolni, hogy mennyi pontjuk van.

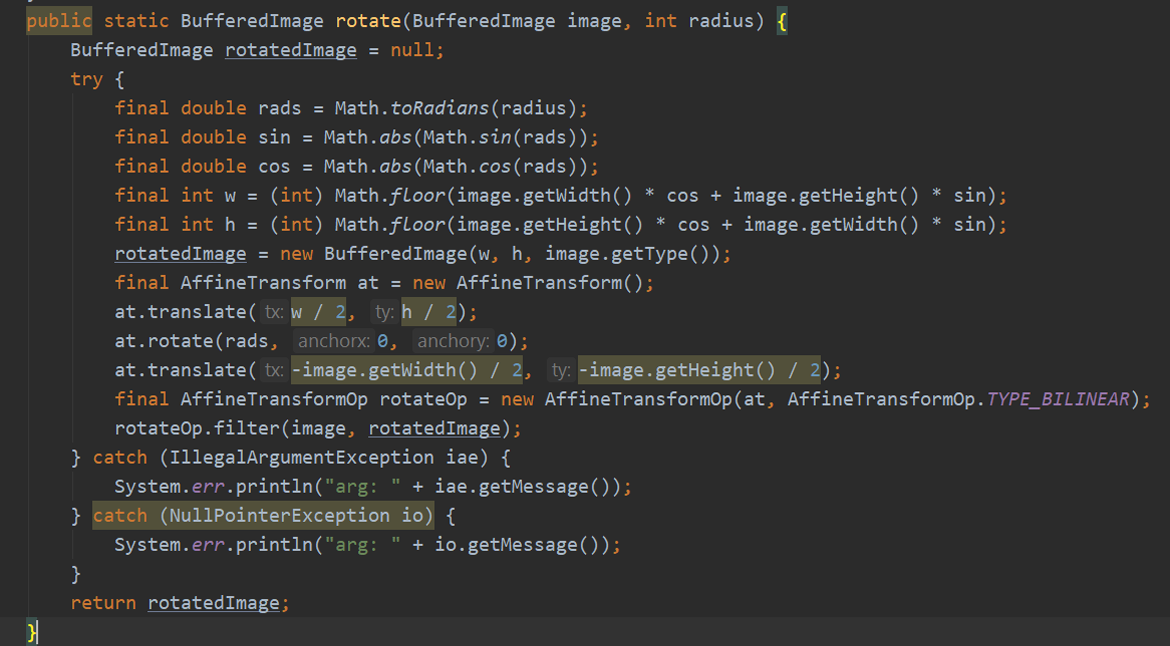
3.Kismars: ilyenkor az a játékos játszik az ellenség ellen, aki bemondta a kismarsot, a csapatársa nem hív lapokat, itt nincsenek fogások. Az a lényege, hogy aki megy kismarsot az elkezdi lehívni a lapjai, a 2 ellenséges játékos sorba rak rá, itt is színre szint kell rakni, de ebben a módban nincs tronf, ha nincs színed bármit rakhatsz, ha tudnak kisebbet rakni úgy, hogy a szín megegyezzen amíg elfogy a lapod akkor ők nyertek, ha nem tudnak akkor te nyertél. Nyerésnél 4 játszmapontot kap a nyert fél.

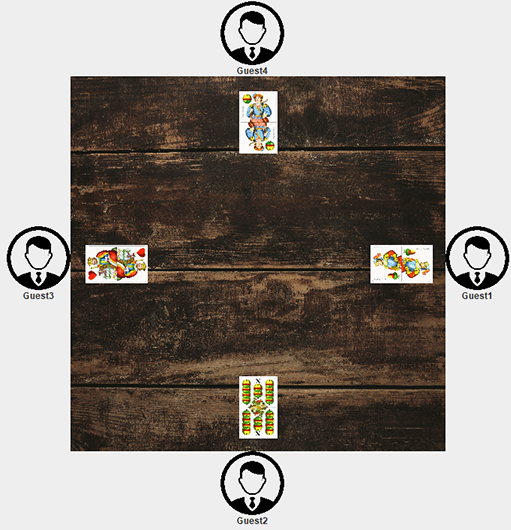
4.Nagymars: ugyan az mint a kismars, csak nagyobbat kell rakni nem kisebbet. Nyerésnél 6 játszmapontot kap a nyert fél.

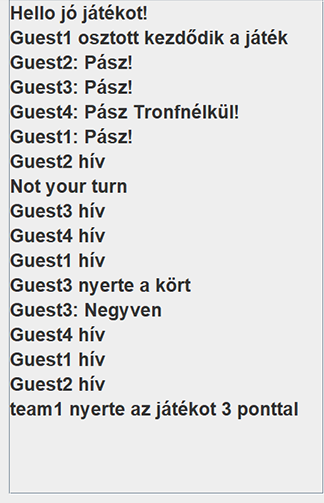
5.Színmars: ezt csak akkor lehet bemondani, ha minden lapod egyforma színű ilyenkor az a csapat, akinek a játékosa bemondta egyből kap 12 pontot és oszt a következő játékos.

Vannak még extra lehetősége a 20 és a 40, a 20 az akkor lehetséges, ha te hívsz és van egy felsőd és egy királyod azonos színből ilyenkor ha lehívod az egyiket és mondod, hogy 20 akkor a fogásaitok összegéhez további 20 pont fog kerülni (nem muszáj ti vigyétek a kört), ugyan így van a negyvennel is csak akkor a felsőd és a királyod tronfból van.  
Ha csak 20- vagy 40-ből van pontotok, akkor érvénytelen, kell minimum egy fogás mellé, hogy érvényesek legyenek.

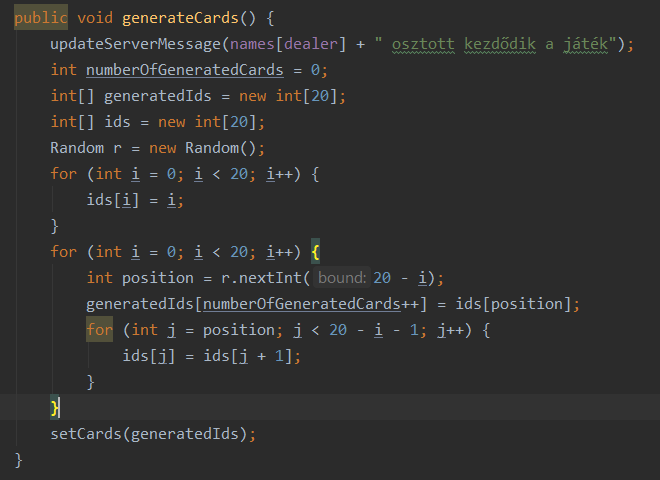
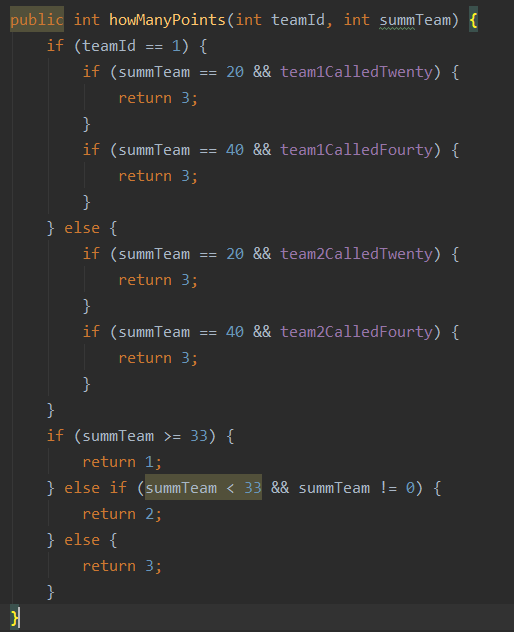
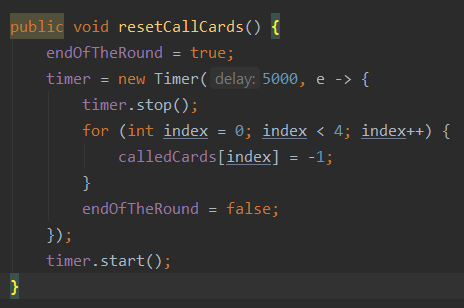
**Project összetétele (14 class)**

1. PngLoader: sima osztály nem terjeszt ki semmit, 2 statikus metódusa van, loadImage és rotate.  
   A loadImage betölti a képet a res könyvtárból, a res könyvtárban található 23 kép (20 kártya és pár grafikai elem).  
   A rotate megfordítja a betöltött képet egy általunk megadott radius szerint. 
2. ImageItem: kiterjeszti a JPanel osztályt a Java Swing GUI-ból, így ez egy panel, amire csak egy megadott képet rajzol ki a loadImage metódus használatával.
3. Card: Panel hasonlóan az ImageItemhez kirajzol egy megadott kártyát a loadImage metódus használatával, csak megjelennek pluszba attribútumok, kártyák színe, azonosítója (id), értéke.
4. Profile: Panel hasonlóan a fent bemutatott példához, amire egy játékos profil kép van kirajzolva az ImageItem segítségével és alatta egy Label-ben a neve.
5. Table: Panel hasonlóan a fent bemutatott példához, amire 4 példány kerül a Profil osztályból és egy ImageItem az asztal képe, majd később a hívott kártyákat is megjeleníti, ezek Card példányok.

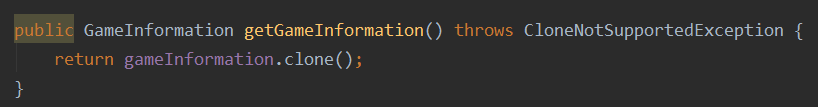


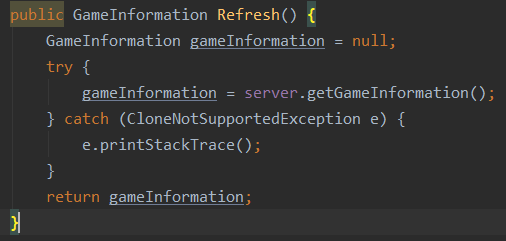
1. Hand: Panel hasonlóan a fent bemutatott példához, van rajta 9 gomb (20,40, osztás, számolás, és a bemondási lehetőségek), és kártyák kirajzolva a Card osztály segítségével, annyi kártya amennyi jelenleg az adott játékos kezében van. A kártyákon és a gombokon Listenerek vannak ezekről majd a project működésénél beszélek. 
2. Chat: Panel hasonlóan a fent bemutatott példához, van rajta 2 TextArea, az egyik írásvédett a másik nem, a felső az a játékosok által küldött üzenetek megjelenítésére szolgál, az alsó az az üzenet küldésére.
3. Logs: Panel hasonlóan a fent bemutatott példához, ami egy TextAreat tartalmaz, ez írásvédett és a játék helyzetet írja le, pl. melyik játékos jön, ki mit mondott be, és azt, hogy mit kell csinálj ha az adott lépéseddel megszegnéd a játék szabályait (minden játékosnak külön logs-ja van).
4. Stats: Panel hasonlóan a fent bemutatott példához, ami a játék statisztikáját tartalmazza, hogy kinek mennyi játszmapontja van és ki hány meccset nyert.
5. Client: kiterjeszti a JFrame osztályt a Java Swing GUI-ból, így ez egy frame, amiben példányosítva vannak a következő panelek: Hand, Table (minden játékosnak más a szemszöge, mindig saját maga van középen), Chat, Logs Stats. Ezek a panelek hozzá vannak adva a framehez.
6. GameInformation: sima osztály nem terjeszt ki semmit, lényege, hogy tartalmazza az összes játékadatot tartalmazza, pl. ki hívott ki nyert, kinek hány lap van a kezébe, mi van asztalon, mi van a chaten, kinek mi van a logs-jában.
7. Server: sima osztály nem terjeszt ki semmit, lényege, hogy tartalmazza a GameInformation osztály egy példányát, és a játék megváltoztatásához szükséges metódusokat.
8. ConnectionToServer: sima osztály nem terjeszt ki semmit, ugyanazokat a metódusokat tartalmazza, mint a Server, továbbiakban fenntartja a kapcsolatot a kliens és a szerver között.
9. Main: sima osztály nem terjeszt ki semmit, lényege, hogy ebben az osztályban példányosítok egy Servert, egy ConnectionToServert, és 4 Client-et.

**Néhány érdekes metódus:**

* **Kártyák generálását végrehajtó metódus a GameInfromation osztályból**:
* **Normál menetben a játék végén eldönti hány pontot fog kapni a nyertes csapat, metódus a GameInformation osztályból**:
* **Kártyák eltűnését végrehajtó metódus a GameInformation osztályból**:

**Projekt Működése, interakciók, játék lefolyása**

* **Kliens-szerver kapcsolat:** Először is a projekt az egy szerver-kliens modell, ami azt jelenti, hogy a szerver szerint frissül a kliens egy adott időközönként, nálam a GamInfromation-ban szereplő adatok szerint a ConnectionToServer osztály segítségével 1 másodpercenként frissül a Client.  
  Nézzük meg ezt a folyamatot: A Server visszaküldi az általa létrehozott GameInformation példányt a gameInformationt, ezt klónozva teszi meg azért, hogy a kliens oldal ne tudjon módosítani ezen a osztályon.



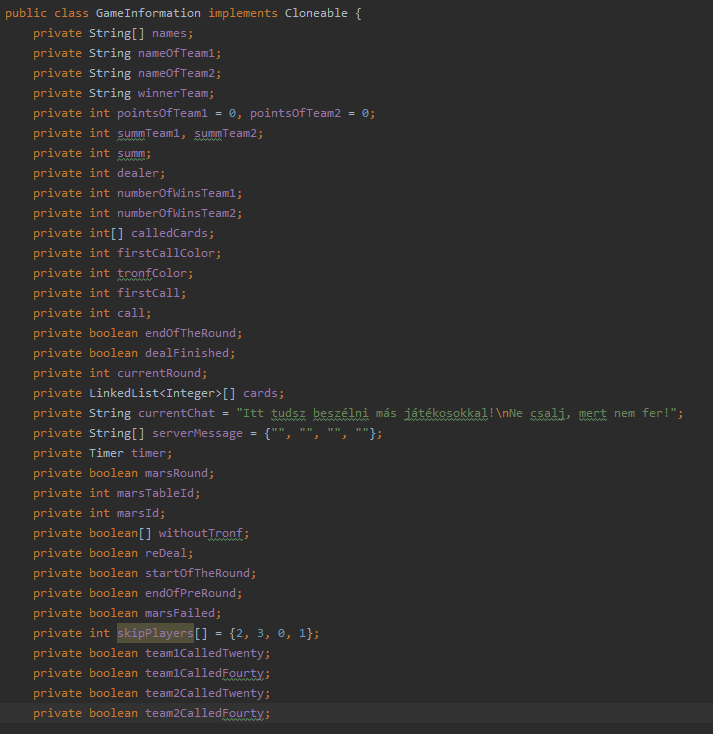
A ConnectionToServer visszaküldi a Server által visszaküldött gameInformation példányt, ha sikerült a klónozás.



A Client 1 másodpercenként megkapja, a ConnectionToServer példányától a connection-től a gameInfromation példányt és ennek segítségével újra rajzólja az egész Client-et.  
Vagyis az a koncepció, hogy a Client ismeri, a ConnectionToServert, és a ConnectionToServer ismeri a Servert. Így jön létre a kliens-szerver kapcsolat.

* **Interakciók:** Kártyák lehívása a kártyákra való kattintással valósul meg, figyelem ha például le van hívva egy szín és van neked abból a színből de másra kattintasz, a játék nem fogja engedni, hogy lehívd, és jobb oldalt, a logs kiírja, hogy a legelőször lehívott színt kell hívj, ha nincs abból és nem tronfra kattintottál, akkor kiírja, hogy tronfot kell hívj.  
  Chat üzenet elküldése az Enter billentyű lenyomásával történik.  
  A bemondásokból csak azok kattinthatóak amik érvényesek, pl ha van tronfod nem kattinthatsz a pász tronfnélkül gombra. Ez érvényes a 20 és a 40 gombokra is. Úgy tudsz 20-at vagy 40-et hívni, hogy rájuk kattintasz így zöldek lesznek, és ezután egy felsőt vagy király hívsz le.  
  A számolás gomb, jobb oldalt a logsba írja ki, hogy mennyi pontotok van a fogásokból.
* **Játék lefolyása:** Miután az osztó oszt egy lap van az asztalon, a tronf. Ezek után a játékosok eldöntik, hogy milyen játékmód van érvényben a bemondásokkal, miután körbeértek , 5 másodperc várakozás után eltűnik a tronf, ez az idő alatt nem tud senki csinálni semmit. Akár milyen játékmódban mikor körbeér a hívás a logs kiírja, hogy ki fog következni, de viszont 5 másodpercig ameddig még láthatóak a kártyák az asztalon nem tud senki csinálni semmit (azt fogja kiírni a logs, hogy nem te következel). Ha megfogtak egy marsban és még nem ért körbe az adott kör akkor, is 5 másodperc után eltűnnek a kártyák hiszen már vége a marsnak, ilyenkor is ki kell várni azt az 5 másodpercet.

GameInfromation osztály attribútumai, ami szerint frissíti magát a Client:



**Fejlesztési lehetőségek**

* A legnagyobb lépés a fejlesztéshez jelenleg az lenne, hogy egy java-ban előre megírt séma segítségével kéne hosztolni a szervert, ez lehetővé tenné az online játékot.
* A logs részletesebb megírása, szebbé tétele, jelenleg van olyan hiba benne, hogy mikor kell várni az 5 másodpercet, hogy eltűnjenek a kártyák akkor azt írja ki annak, aki pont következőkben kéne hívjon, hogy “nem te jössz”.
* Több lobby létrehozása szóval egyszerre, nem csak 4 játékos tud játszani, hanem sokkal több párhuzamos játék, de ehhez szükséges, hogy online fusson a játék.
* Játékosok által írt üzenetek megjelenítése az asztalon az ikonok mellett.  
  egy buborékban, játékosok tudjanak emotokat, hangulatjeleket megjeleníteni.
* Játékos statisztika, megőrzése, megjelenítése, hány kört nyertél, hány kört vesztettél, mennyi marsot nyertél, mennyit buktál, összes győzelmek száma. Ehhez szükséges lenne egy menü létrehozása.
* Több nyelven való játék, nyelv változtatási lehetőség.

**Végszó**

A Snapszer játékot legfőképpen azért készítettem el, mert az iskolában mindig ezt játszódjuk, és kíváncsi lettem mennyi idő lenne megírni. Az első lépés a játék elmélete és a kommunikáció 1-2 nap alatt megvolt így már lehetett játszani, de azt tanácsolták, hogy átkéne írjam kliens-szerver modellbe. Innentől újra kellet kezdjem az egészet szóval nagyjából 1-1,5 hónapot vett igénybe a játék elkészítése a jelenlegi verzióig. Sajnos az online részét már magamtól nem tudom megcsinálni, de megpróbálok segítséget kérni olyan tanároktól/diákoktól akik értenek hozzá. A kliens-szerver modell megírása elég mély tudást igényel az Objektumorientált programozás (OOP) terén, szerencsére nekem nem voltak nehézségeim a játék elemek összekapcsolásában, gyakorlásnak ez a kicsi kártyajáték tökéletes volt szerintem.

**Köszönöm a figyelmet!**A játék publikus elérhető a[**https://gitlab.com/DMozes/snapszer**](https://gitlab.com/DMozes/snapszer)**.**