A feladat teszteléséhez a mellékelt projekt tartalmaz egy *main* függvényt. Minden más osztályt a feladat során kell megírni. A feladat során egy számítógépes játék karaktereit kell kezelni, akiket három kasztra lehet osztani: harcos, íjász, és mágus.

- Töltsd le, és add a projekthez a **json.hpp** fájlt.
- Készíts egy **Karakter** osztályt, ami egy absztrakt ősosztály lesz a különböző típusú karakterekhez. Az osztály tárolja a karakter nevét (szöveg) és szintjét (egész). Mindkét adatot a konstruktor várja, és legyen hozzájuk getter (*getNev*, *getSzint*).
- Legyen a Karakter osztálynak három tisztán virtuális metódusa:
  - *eletero*: a karakter életerejét adja majd vissza egész számként, az adatokon nem módosít,
  - *sebzes*: a karakter sebzését adja majd vissza egész számként, az adatokon nem módosít,
  - *kiir*: megjeleníti majd a karakter adatait, és nem módosít rajtuk.
- Származtass a **Karakter** osztályból egy **Harcos** osztályt, ami a harcos típusú karaktert valósítja meg. Extra adatként a használt fegyver típusát tárolja (szöveg). A konstruktor mindhárom adatot várja (név, szint, fegyver).
- Fejtsd ki a **Harcos** osztályban az ősosztály metódusait az alábbiaknak megfelelően:
  - *eletero*: a harcos életereje alapból 12, és ez minden egyes szintért 6-tal növekszik,
  - *sebzes*: a harcos sebzése alapból 5, és ez minden egyes szintért 2-vel növekszik,
  - *kiir*: megjeleníti az adatokat a minta szerint.
- Származtass a **Karakter** osztályból egy **Ijasz** osztályt, ami az íjász típusú karaktert valósítja meg. Extra adatként a célzás pontosságát tárolja (egész). A konstruktor mindhárom adatot várja (név, szint, pontosság).
- Fejtsd ki az **Ijasz** osztályban az ősosztály metódusait az alábbiaknak megfelelően:
  - *eletero*: az íjász életereje alapból 10, és ez minden egyes szintért 3-mal növekszik,
  - *sebzes*: az íjász sebzése alapból 3, és ez minden egyes szintért 2-vel növekszik, azonban, ha a karakter pontossága legalább 90, akkor a sebzés duplázódik,
  - *kiir*: megjeleníti az adatokat a minta szerint.
- Származtass a **Karakter** osztályból egy **Magus** osztályt, ami a mágus típusú karaktert valósítja meg. Extra adatként a varázsláshoz rendelkezésre álló mana mennyiségét tárolja (egész). A konstruktor mindhárom adatot várja (név, szint, mana).
- Fejtsd ki a **Magus** osztályban az ősosztály metódusait az alábbiaknak megfelelően:
  - *eletero*: a mágus életereje alapból 8, és ez minden egyes szintért 3-mal növekszik, plusz 1 extra életet kap minden 5. manáért (törtrész nincs),
  - *sebzes*: a mágus sebzése alapból 4, és ez minden egyes szintért 2-vel növekszik, plusz 3 extra sebzést kap minden 10. manáért (törtrész nincs),
  - *kiir*: megjeleníti az adatokat a minta szerint.
- Készíts egy **Szerver** osztályt, ami a játékteret valósítja meg. A szerver a karakterek tömbjét tárolja (célszerű **vector** segítségével, de nem kötelező). Mivel minden fajta karaktert egyben kell kezelni, értelemszerűen **Karakter** mutatók tömbjét kell tárolni.

- A **Szerver** osztálynak legyen egy *betolt* metódusa, amely a **karakterek.json** fájlból betölti a szerveren játszó játékosokat. A fájl JSON formátumú, egy tömböt tartalmaz, amiben a karakterek találhatóak. Minden karakterhez adott a neve, kasztja, szintje, valamint a kaszthoz tartozó extra adat (fegyver, pontosság, vagy mana). A metódus töltse fel a tárolt karakterek tömbjét a megfelelő típusú karakterekkel.
- A **Szerver** osztálynak legyen egy *listaz* metódusa, amely kilistázza a játékosokat (a **Karakter** osztály *kiir* metódusát felhasználva).
- A **Szerver** osztály destruktora szabadítson fel minden lefoglalt memóriát.
- A **Szerver** osztálynak legyen egy *legtobbElet* metódusa, amely visszaadja a legtöbb élettel rendelkező karaktert (**Karakter** mutatóként, értelemszerűen).
- Legyen a **Szerver** osztálynak egy *csata* metódusa, amely megkapja kettő játékos nevét (szövegek), eldönti, hogy ha ez a két játékos összecsap, akkor ki lesz a nyertes, és ennek megfelelően visszaadja a nyertes nevét (szöveg). Feltehetjük, hogy létező játékos neveket adunk meg. Ha a csata döntetlenre jönne ki, akkor az „X” szöveget adja vissza.
    - Hogy megy a csata? Mindkét játékosnak van egy életereje és egy sebzése. A csata minden körében a játékosok a sebzésüknek megfelelően csökkentik a másik játékos életét. Ha valaki élete 0-ra, vagy ez alá csökken, akkor a másik nyer. Ha egyszerre csökken 0-ra (vagy az alá) az életük, akkor döntetlen. Ha mindketten élnek még, akkor jön a következő kör. Például az egyik játékos 100 élettel és 40 sebzéssel rendelkezik, a másik 78 élettel és 48 sebzéssel. Az első kör után az első játékosnak 100-48=52 élete marad, a második játékosnak 78-40=38. A második kör után az életek 4 és -2, vagyis az első játékos nyert.
    - Értelemszerűen a játékosok tárolt adatai nem változhatnak, az „élet csökkenést” átmeneti adatokkal kell kezelni.
- Legyen a **Szerver** osztálynak egy *tornaSzimulacio* metódusa, amely több csatát is végrehajt (az előző *csata* függvényt felhasználva). A lejátszandó csatákat a **merkozesek.json** fájl tárolja JSON formátumban. Egy tömböt tartalmaz, aminek minden eleme két adatból áll: **jatekos1** és **jatekos2**, a két játékos neve. A függvény mindegyik csatát futtassa le, majd az eredményt írja ki az **eredmenyek.json** fájlba, JSON formátumban. A kimeneten ugyanaz az adatszerkezet legyen, mint amit be is olvas, de a tömb minden eleme egészüljön ki egy **nyertes** adattal, ami 1, ha az első játékos nyert, 2, ha a második, és 0, ha döntetlen.