## Háziorvos rendelési idő – 8 pont

A háziorvosa reggel nyolc órakor kezdi a rendelést, és délben fejezi be – azaz 8:00‑kor már rendel és 12:00-kor már nem fogad új beteget.

Írjon programot **rendeles.py** néven! A program kérjen be egy egész órát jelző számot a felhasználótól, majd írja ki, hogy a megadott időpontban rendel-e a háziorvosa! Amennyiben igen, akkor azt is írja ki, hogy hány perc van még a rendelési idő végéig (óra száma\*60), azaz hány perc áll a rendelkezésre odaérni az orvosi rendelőbe! Ha még nem rendel, azaz még nincs 8 óra, akkor írja ki, hogy hány perc múlva kezdődik a rendelés, vagy sürgős esetben pedig hívja az ügyeletet! Ha már nem rendel, akkor írja ki: „Ma már nem rendel a doktorúr, keresse holnap reggel, sürgős estben hívja az ügyeletet!” A program üzeneteinek megfogalmazásában kövesse az alábbi példát! Azokat a részeket, amiket a felhasználó gépel be, a mintában vastagított és döntött betűkkel emeltük ki.

PS c:/python/haziorvos.py

Hány óra van most? ***7***

Még nem rendel a doktorúr, sürgős esetben hívja az ügyeletet, vagy ennyi idő múlva keresse a rendelőben: 1

PS c:/python/haziorvos.py

Hány óra van most? ***8***

Ennyi perced van még odaérni: 240

c:/python/haziorvos.py

Hány óra van most? ***11***

Ennyi perced van még odaérni: 60

c:/python/haziorvos.py

Hány óra van most? ***13***

Ma már nem rendel a doktorúr, keresse holnap reggel, sürgős estben hívja az ügyeletet!

#### Pontozás – minden teljesülő feltétel egy-egy pontot ér

1. Létrehoz programot rendeles.py néven, a program hibaüzenet nélkül lefut.
2. Bekéri a felhasználótól az óra értéket, és tárolja.
3. A bekért értéket szám típusúvá alakítja.
4. Elágazást használ a különböző esetek kezelésére.
5. Helyesen állapítja meg és írja ki, ha még nincs rendelés és mikor lesz.
6. Helyesen állapítja meg és írja ki, hogy van rendelés és még hány percig.
7. Helyesen állapítja meg és írja ki, ha nincs rendelés.
8. A kiírt üzenetek helyesek (pl.: nincs benne elgépelés, helyesen jelennek meg a szóközök).

## feladat Számolás – 14 pont

A program feladata a négy alapművelet elvégzése függvények segítségével.

1. Írjon programot **szamol.py** néven!
2. Írja meg a muveletek.py állományban található függvények hiányzó műveleteit!
3. A programban használja a muveletek.py állomány ön által kiegészített függvényeit. A visszatérési érték az elvégzett művelet eredménye legyen. A kiegészített függvényt használja fel a programjában (bemásolhatja).
4. Kérjen be két számot és egy műveleti jelet (+ - \* / )
5. A bekért számokkal végezze el a beolvasott műveletet. Ehhez hívja meg a létrehozott függvényeket, meghíváskor a függvény paramétere minden művelet esetén a két szám legyen.
6. Csak azt a műveletet kell elvégezni, amelyiket a számokhoz beolvasott.
7. A program négyszer egymás után kérjen be számpárt és műveleti jelet, majd végezze el a műveletet és írja ki az eredményt.
8. A program üzeneteinek megfogalmazásában kövesse az alábbi példát! Azokat a részeket, amiket a felhasználó gépel be, a mintában vastagított és döntött betűkkel emeltük ki.

#### PS c:/python/szamol.py

#### Kérem az első számot!**3**

#### Kérem az első számot!**4**

#### Milyen műveletet végezzek el (+ / - \*)? \*

#### 3 \* 4 = 12

#### Kérem az első számot!**5**

#### Kérem az első számot!**6**

#### Milyen műveletet végezzek el (+ / - \*)? +

#### 5 + 6 = 11

#### Kérem az első számot!**1**

#### Kérem az első számot!**2**

#### Milyen műveletet végezzek el (+ / - \*)? /

#### 1 / 2 = 0.5

#### Kérem az első számot!**7**

#### Kérem az első számot!**9**

#### Milyen műveletet végezzek el (+ / - \*)? -

#### 7 - 9 = -2

#### Pontozás – minden teljesülő feltétel egy-egy pontot ér

1. Létrehoz programot szamol.py néven, a program hibaüzenet nélkül lefut.
2. Bekér egy számot és tárolja.
3. A bekért számot számmá alakítja
4. Bekéri a másik számot is és számmá alakítja
5. Bekéri a műveleti jelet a mintában szereplő kiírással
6. 4 lépéses ciklust szervez számok és a műveleti jel bekérésére, illetve a műveletek elvégzésére.
7. Kiegészíti a függvényeket a hiányzó műveletekkel
8. A függvényt bemásolja, vagy hivatkozik rá a szamol.py programban
9. A függvényt a két szám paraméterrel hívja meg
10. Legalább egy műveleti jel alapján meghívja a megfelelő függvényt
11. Mind a négy művelet esetén a megfelelő függvényt hívja meg
12. A függvény visszatérési értéke alapján a főprogram (vagy az annak megfelelő függvény) írja ki a műveletet és az eredményt.
13. A függvényhívás a ciklusmagba kerül.
14. A kiírt üzenetek helyesek (pl.: nincs benne elgépelés, helyesen jelennek meg a szóközök).

## feladat Elektromos autók teljesítménye – 18 pont

Az elkészítendő program elektromos autók nevét és az általuk egy felöltéssel megtehető távolságot tárolja. Kérje be a felhasználótól három autó nevét és a teljesíteni képes távolság értékét, majd ezt követően határozza meg és írja ki fájlba a legnagyobb teljesítményre képes autó nevét.

1. Írjon programot **legtovabb.py** néven!
2. Az autók adatainak tárolására szolgáló objektumok alapját képező osztály rendelkezésre áll az eautok.py fájlban. A programjában töltse be ezt a modult, és használja a bene lévő osztályt!
3. Kérje be a felhasználótól három autó nevét, teljesítményét kilométerben! A beolvasott adatok alapján hozza létre az Eauto osztályú objektumokat és tárolja őket!
4. Határozza meg a legnagyobb teljesítményt! Feltételezheti, hogy a felhasználó nem ad meg egyező teljesítmény adatokat.
5. Határozza meg, hogy ez a teljesítmény melyik autóhoz tartozik, és a nevét írja ki a „legtovabb.txt” szövegfájlba!

A program üzeneteinek megfogalmazásában kövesse az alábbi példát! Azokat a részeket, amiket a felhasználó gépel be, a mintában vastagított és döntött betűkkel emeltük ki. A feladat hibátlan elvégzéséért 18 pont jár.

#### c:/python/legtovabb.py

#### Adja meg az elektromos autó nevét! **Tesla**

#### Hány km-t tud megtenni egy feltöltéssel? **368**

#### Adja meg az elektromos autó nevét! **E-tron**

#### Hány km-t tud megtenni egy feltöltéssel? **236**

#### Adja meg az elektromos autó nevét! **Zoe**

#### Hány km-t tud megtenni egy feltöltéssel? **248**

#### A(z) Tesla 368 km-t tud megtenni egy feltöltéssel

#### A(z) E-tron 23 km-t tud megtenni egy feltöltéssel

#### A(z) Zoe 248 km-t tud megtenni egy feltöltéssel

#### Az eauto.py modul tartalma

class Eauto:

    def \_\_init\_\_(self,nev,km):

        self.nev=nev

        self.km=km

#### Pontozás – minden teljesülő feltétel egy-egy pontot ér

1. Létrehoz programot legtovabb.py néven, a program hibaüzenet nélkül lefut.
2. Importálta a mellékelt ***eauto*** osztályt tartalmazó modult
3. Bekéri egy autó nevét.
4. Bekéri egy autó km teljesítményét.
5. A km értéket egész számmá alakítja.
6. A név és km adatokból ***Eauto*** osztályú objektumot hoz létre.
7. Egy listát létrehoz az autók adatainak tárolására
8. Egy ***Eauto*** osztályú objektumot elhelyez a létrehozott adatszerkezetben.
9. Három bekért név, km alapján három objektumot hoz létre.
10. Mindhárom objektumot elhelyezi az adatszerkezetben.
11. Az adatszerkezetben tárolt objektumok alapján megjelenít egy objektumot” NEV km -t tud megtenni egy feltöltéssel” formában a minta alapján.
12. Mindhárom objektumot megjeleníti.
13. A legtovább jutó kiválasztása kezdetén az első elemet elhelyezi a legtovább jutó elemet tároló változóban.
14. A tároló elemeinek vizsgálatára ciklust hoz létre
15. Összehasonlítja az aktuális elemet a tárolt legnagyobb elemmel
16. Ha nagyobbat talál, akkor felülírja a legnagyobb elemet, ha kisebb az értéke, akkor nem történik változás
17. Létrehozta írásra a legtovabb.txt állományt.
18. Az legtavolabb.txt állományban csak a legtávolabbra jutó autó neve található.