Feladatokra vonatkozó általános elvárások

- *Az Ön feladata az alábbiakban olvasható leírás alapján három program elkészítése.*
- *A három Python-feladat elvégzésére összesen 60 perc áll rendelkezésre.*
- *A programokat a (megadott hely)-re kell mentenie.*
- *A programok elkészítése során a felhasználó által megadott adatok helyességét nem kell ellenőriznie – ha például a program egy 1 és 5 közé eső szám megadását kéri a felhasználótól, akkor feltételezheti, hogy a felhasználó számot, és a megadott feltételeknek megfelelő számot ad meg.*
- *Törekedjen arra, hogy a tanult programozási elveknek megfelelő adatszerkezeteket, vezérlési szerkezeteket alkalmazzon!*
- *Munkáját rendszeresen mentse! Amennyiben majd a vizsga során a számítógép nem megfelelő működését tapasztalja, jelezze a felügyelő tanárnak!*

Első feladat
Almafarm

Írjon programot **almafarm.py** néven! A program kérjen be két adatot a felhasználótól: hány rekesz almát rendeltek a farmtól (5 és 20 között lehet) és az adott napon hány darab almát szüreteltek le (100 és 200 között lehet). Egy rekeszbe 12 alma fér bele.

Írassa ki, hogy tudjuk-e teljesíteni a rendelt mennyiséget, azaz minden rendelt rekeszt tele tudtunk-e tölteni! Ha nem szedtek elég almát, akkor azt írassa ki, hogy hány rekeszt tudnak aznap értékesíteni!

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 összesen 8 pont jár.

C:\Users\raerek\programok>***almafarm.py***
Adja meg a rendelt rekeszek darabszámát (5-20): ***8***
Adja meg a mai napon leszüretelt almák darabszámot (100-200): ***112***
A rendelt mennyiség teljesíthető.
C:\Users\raerek\programok>***almafarm.py***
Adja meg a rendelt rekeszek darabszámát (5-20): ***10***
Adja meg a mai napon leszüretelt almák darabszámot (100-200): ***112***
A rendelt mennyiség nem teljesíthető, max. 9 rekeszt lehet értékesíteni.

Pontozás – minden teljesülő feltétel egy-egy pontot ér

1. Létrehoz programot almafarm.py néven, a program hibaüzenet nélkül lefut.
2. Bekéri a felhasználótól az egyik számot, és tárolja.
3. A bekért számot szám típusúvá alakítja.
4. Az előző két lépést a második számmal is elvégzi.
5. Elágazást használ a különböző esetek kezelésére.
6. Ha a rendelt rekeszek számának és az egy rekeszbe tehető almák számának a szorzata kisebb vagy egyenlő, mint a leszedett almák száma, akkor helyesen állapítja meg és írja ki, hogy a rendelt mennyiség teljesíthető.
7. Ha a rendelt rekeszek számának és az egy rekeszbe tehető almák számának a szorzata nagyobb, mint a leszedett almák száma, akkor helyesen állapítja meg és írja ki, hogy a rendelt mennyiség nem teljesíthető és helyesen jeleníti meg, hogy hány rekeszt lehet maximum értékesíteni.
8. A kiírt üzenetek helyesek (pl.: helyesen jelennek meg a szóközök).

Második feladat: Bank

A programot az XY bank, az általa népszerűsíteni kívánt befektetési forma prezentálásához rendelte meg.
A program a betétek kezdeti összegéből és a futamidőjükből (hónapban megadva) eldönti és kiírja, hogy a leendő betétesek mekkora összegű betéti kamatot fognak a futamidő végén kapni. A kapott kamat összege a betett betét és az akciós kamat (12%-os) szorzata lesz, de csak akkor (csak ebben az esetben jár az akciós kamat), ha a futamidő több mint 12 hónap. Egyéb esetben nem jogosult az leendő betétes az akciós betéti kamatra, ebben az esetben csak 5%-os betéti kamatra jogosult.
(A programban nem kell a kamatot időarányosan számolni. Ugyanannyi kamat jár 13 hónapra is, mint 24 hónapra. A kapott betéti kamat összegét 2 tizedesjegy pontosan kell a képernyőn megjeleníteni.)

Írjon programot **bank.py** néven!

Kérjen be 4 kezdeti összeget és 4 futamidőt! Írja meg azt a függvényt, ami eldönti, hogy a betenni kívánt betét után mekkora összegű betéti kamat kapható! A függvény paraméterei a kezdeti összeg és a futamidő legyen, a visszatérési értéke pedig számérték legyen: a kapott betéti kamat összege. Ezt a függvényt használja fel a programjában!

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 összesen 14 pont jár.

```
C:\Users\raerek\programok>bank.py
Adja meg a betenni kívánt összeget: 5000
Adja meg betét futamidejét (hónapban): 10
A betett összeg után 250.00 Ft betéti kamat jár.
Adja meg a betenni kívánt összeget: 5000
Adja meg betét futamidejét (hónapban): 12
A betett összeg után 250.00 Ft betéti kamat jár.
Adja meg a betenni kívánt összeget: 5000
Adja meg betét futamidejét (hónapban): 13
A betett összeg után 600.00 Ft betéti kamat jár.
Adja meg a betenni kívánt összeget: 5000
Adja meg betét futamidejét (hónapban): 24
A betett összeg után 600.00 Ft betéti kamat jár.
```

Pontozás – minden teljesülő feltétel egy-egy pontot ér

1. Létrehoz programot bank.py néven, a program hibaüzenet nélkül lefut.
2. Bekér egy kezdeti összeget és tárolja.
3. A bekért összeget szám típusúvá alakítja.
4. Az előző két lépést a második számmal is elvégzi.
5. Egy esetben a kapott adatok alapján helyesen állapítja meg a kapható betéti kamat összegét.
6. Egy esetben helyesen jelenít meg üzenetet a kapható betéti kamat összegéről.
7. Ciklust szervez az adatok bekérésére, illetve a kapható betéti kamat összegének kiírására.
8. A ciklus futása 4 adatpár megadása után véget ér.
9. Függvényt hozott létre a kapható betéti kamat összegének megállapítására.
10. A függvény paramétere a kezdeti összeg és a futamidő.
11. A függvényt helyesen hívja meg.
12. A függvény visszatérési értéke alapján a főprogram írja ki a kapható betéti kamat összegét.
13. A függvényhívás a ciklusmagba kerül.
14. A kiírt üzenetek helyesek (pl.: helyesen jelennek meg a szóközök, a kapott számértékek 2 tizedesjegy pontosak).

Harmadik feladat
Sakkolimpia 2040

Írjon programot **sakk2040.py** néven! Az elkészítendő program kiinduló állománya (***eredmenyek.txt***) sakkpartik eredményeit tartalmazza egymás alatt a 2040 olimpián. Egy sorban, egy parti végeredménye, azaz annak a rövidítése, hogy melyik színben játszó játékos nyert („FH” illetve „FK”) szerepel, vagy az „XX”, ha az adott parti végeredménye döntetlen lett. Ezeket egy megfelelő adatszerkezetbe tárolja le! Majd ezt követően határozza meg és írassa ki a ***pontozotabla.txt*** állományba, hogy összesen hány partit játszottak az olimpián, melyik színnel játszva hányszor nyertek és hányszor lett döntetlen a partik végeredménye! A fájlba való kiíratás után (melynek sikeres lefutását most nem kell vizsgálnia) tájékoztassa a felhasználót a kiíratás megtörténtéről!

A program üzeneteinek megfogalmazásában kövesse az alábbi példát!

A feladat hibátlan elvégzéséért összesen 18 pont jár.

C:\Users\raerek\programok>***tippmix.py***
Az állományba történt kiíratás megtörtént.

A pontozotabla.txt tartalma (minta):
A 2040-es sakkolimpián a következő eredmények születtek:
Az olimpián összesen 83 partit játszottak a versenyzők egymással.
Ezek lebontása a következő:
Fehér színben játszva összesen 32 partit nyertek.
Fekete színben játszva összesen 30 partit nyertek.
Döntetlent pedig összesen 21 esetben játszottak.

Pontozás – minden teljesülő feltétel egy-egy pontot ér

1. Létrehoz programot sakk2040.py néven, a program hibaüzenet nélkül lefut.
2. Létrehozz egy adatszerkezetet, az adatok tárolására.
3. Megnyitja olvasásra az eredmenyek.txt fájlt.
4. Letárol az adatszerkezetben 1 adatot.
5. Letárolja a fájlban található összes adatot.
6. Gondoskodik a fájl olvasás utáni bezárásáról.
7. Megállapítja, hogy hány partit játszottak összesen.
8. Megállapítja, hogy hány partit nyertek fehér színben (FH) játszva.
9. Megállapítja, hogy hány partit nyertek fekete színben (FK) játszva.
10. Megállapítja, hogy hány parti végeredménye lett döntetlen.
11. Megnyitja a pontozotabla.txt fájlt írásra.
12. Kiíratja, hogy hány partit játszottak összesen.
13. Kiíratja, hogy hány partit nyertek fehér színben (FH) játszva.
14. Kiíratja, hogy hány partit nyertek fekete színben (FK) játszva.
15. Kiíratja, hogy hány parti végeredménye lett döntetlen.
16. Gondoskodik a fájl írása után a bezárásáról.
17. A képernyőn tájékoztatja a felhasználót a fájlba történt kiíratás megtörténtéről.
18. A kiírt üzenetek helyesek (pl.: helyesen jelennek meg a szóközök).