Viikkotehtävät 4

Merkkijonojen käsittely, virheenkäsittely, Boolean-muuttujat

1. Tee ohjelma, jossa on seuraavat tiedot omissa muuttujissaan (älä kysy käyttäjältä tietoja input():lla, vaan aseta suoraan muuttujiin haluamasi arvot koodissa!):

   - **Päivämäärä (teksti)**
   - **Asiakkaan nimi (teksti)**
   - **Syntymävuosi (luku)**
   - **Saldo (luku)**

   Yhdistä muuttujat uuteen muuttujaan siten, että voit tulostaa näytölle tiedot seuraavalla tavalla, esimerkki (muuttujien arvot merkitty punaisella):

   Matti Meikäläinen (1984), saldo: 345 €, päivitetty 1.10.2019.

   Esimerkki ohjelman toiminnasta:

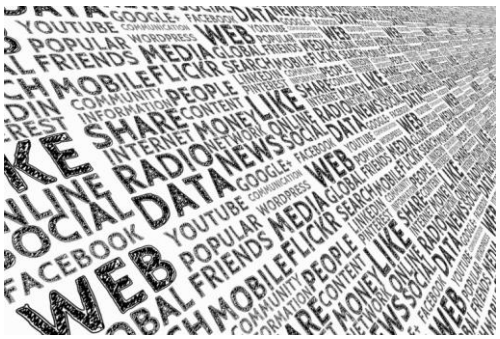
   ```
   Testi Henkilöinen (1982), saldo: 1698 €, päivitetty 12.8.2020.
   ```

   Ehdottomasti suositeltavin tapa on hyödyntää f-stringiä tässä tehtävässä!

   HUOM!
   Ole erityisen tarkkana välilyöntien ja erikoismerkkien kanssa, codePostin palautuslaatikko vaatii tekstin täsmälleen samassa formaatissa kuin yllä olevassa esimerkissä!
   Tämä on tehty tarkoituksella, sillä myös työelämässä asiakkaat ovat joskus pilkuntarkkoja tällaisten yksityiskohtien kanssa. On parempi oppia heti alussa jopa liian tarkaksi yksityiskohtien osalta. ☺

   Pieni lisätehtävä: Hae päivämäärä ***datetime***-moduulista! (ks. Osa 1 -materiaaleista)
   Huom: älä tulosta pelkkää **date**-muuttujan aikaleimaa, vaan käytä myös **strftime()**-funktiota, jotta voit tulostaa päivämäärän muodossa **PP.KK.VVVV**, ks. materiaalit / luento 2

   Tehtävän tiedostonimi = ***exercise4_1.py***
   Tyypillinen koodimäärä***: 5-8 riviä*** *(tyhjiä rivejä ja kommentteja ei lasketa)*



2. **Tee ohjelma, joka kysyy käyttäjältä tekstiä muuttujaan.**
   - Käännä teksti toisinpäin, ja tulosta se näytölle.

   - Testaa myös, onko annettu teksti palindromi. (jos teksti on oikeinpäin ja väärinpäin identtinen, on kyseessä palindromi. Esimerkiksi ”saippuakauppias”.)

     Jos sana on palindromi, tulosta **Palindromi: KYLLÄ**, jos ei, tulosta **Palindromi: EI**.

   - Jos käyttäjä syöttää tyhjän tekstin, tulosta **ainoastaan** ”Antamasi teksti on tyhjä.”

Esimerkkejä ohjelman toiminnasta:

```
Anna jokin teksti:
melkoinenlause
esualnenioklem
Palindromi: EI
```

```
Anna jokin teksti:
saippuakauppias
saippuakauppias
Palindromi: KYLLÄ
```

```
Anna jokin teksti:


Antamasi teksti on tyhjä.
```

```
Anna jokin teksti:
niin niin
niin niin
Palindromi: KYLLÄ
```

Vinkki: Noudata samoja sanamuotoja kuin esimerkeissä, **codePost** ei muuten välttämättä tunnista miten ohjelmasi toimii!

Tehtävän tiedostonimi = ***exercise4_2.py***

Tyypillinen koodimäärä***: 7-10 riviä*** *(tyhjiä rivejä ja kommentteja ei lasketa)*

3. Harjoitellaan tyypillisiä tekstioperaatioita!
Tallenna seuraava teksti muuttujaan nimeltä **text**, ja tulosta se näytölle **(koodista myös helpommin kopioitava versio Moodlessa!)**:

```
text = "The quick brown fox jumps over the lazy dog. That \
sentence contains every letter in the English alphabet. Neat!"

print(text)
```

Huom: Älä tee tässä tehtävässä aina uutta muuttujaa tekstille, vaan muokkaa aina olemassa olevaa **text**-muuttujaa! (muutoin lopputulos ei välttämättä täsmää kohdissa c) ja d))

a) Muokkaa **text**-muuttujaa niin, että ”fox” vaihdetaan sanaan ”duck”. **Tulosta muokattu text-muuttuja näytölle uudestaan**.
- **Huom:** muuta **text**-muuttujaa suoraan, älä tee uutta muuttujaa!
(**esim. ei: ~~new_text = text.replace(…~~**) …vaan **text = text.replace(…. jne.**

b) Pyydä käyttäjältä jokin sana input():lla, joka poistetaan **text**-muuttujasta.
- **Jos** sanaa ei löydy **text**-muuttujasta, tulosta ”Sanaa ei löytynyt.”
- **Jos** poistettava sana löytyi **text**-muuttujasta, poista sana **text**-muuttujasta ja tulosta **text**-muuttuja uudestaan näytölle
  - **Huom:** muuta **text**-muuttujaa suoraan, älä tee uutta muuttujaa!
  Esim. **text = text.replace(…. jne.**

Vinkki: poistaminen tapahtuu korvaamalla poistettava sana tyhjällä tekstillä: ””

c) Tulosta ***text**-muuttujassa* olevien merkkien lukumäärä tällä hetkellä.
Pieni lisätehtävä: Tulosta näytölle myös **text**-muuttujan sanojen lukumäärä.

d) Muuta **text**-muuttujan tekstin pisteet rivinvaihdoiksi. Tulosta **text**-muuttuja uudestaan näytölle.
- **Huom:** muuta **text**-muuttujaa suoraan, älä tee uutta muuttujaa!
Esim. **text = text.replace(…. jne.**

e) Pyydä käyttäjältä jokin lause, ja tallenna se **input()**:n avulla muuttujaan nimeltä **usertext**
- Tee **if**-lause, ja jos **usertext** on 20 tai alle merkkiä pitkä, tulosta **usertext** sellaisenaan
- …mutta jos **usertext** on yli 20 merkkiä pitkä, lyhennä se tasan 20 merkkiä pitkäksi, sekä lisää perään kolme pistettä "…". Tulosta lyhennetty versio **usertext**-muuttujasta tämän jälkeen

Esim. jos käyttäjä syöttäisi lauseeksi "*Tämä on todella pitkä testilause*", olisi lopputulos tämä:

```
Tämä on todella pitk...
```

Esimerkki ohjelman toiminnasta (sivuttaissuunnassa näkyy vain osa):

```
The quick brown fox jumps over the lazy dog. That se

The quick brown duck jumps over the lazy dog. That s

Anna poistettava sana:
quick

Merkkejä: 107

The  brown duck jumps over the lazy dog
 That sentence contains every letter in the English
 Neat!

Anna jokin lause:
Rovaniemi sijaitsee napapiirin tuntumassa.
Rovaniemi sijaitsee ...
```

Huom! Älä käytä exit() –funktiota tässä tehtävässä!

Tehtävän tiedostonimi = ***exercise4_3.py***

Tyypillinen koodimäärä***: 18-28 riviä*** *(tyhjiä rivejä ja kommentteja ei lasketa)*

4. Pyydä käyttäjältä kaksi eri numeroa. Tulosta lopputuloksena jakolaskun tulos, jossa jaat ensimmäisen luvun toisella luvulla. Jos käyttäjä syöttää tekstiä, tulosta näytölle ainoastaan ”Virheellinen muoto.” Jos käyttäjä yrittää jakaa nollalla, tulosta näytölle ainoastaan ”Nollalla ei saa jakaa.” **Käytä ohjelmassa virheenkäsittelyä (try/except)**.'

   Noudata tehtävässä samoja tekstiulkoasuja kuin alla olevissa kuvissa.

   Esimerkkejä ohjelman toiminnasta:

```
Anna ensimmäinen numero:
5
Anna toinen numero:
3
1.6666666666666667
```

```
Anna ensimmäinen numero:
terve!
Virheellinen muoto.
```

```
Anna ensimmäinen numero:
12
Anna toinen numero:
0
Nollalla ei saa jakaa.
```

   Vinkki: Älä tee virheiden tarkistamista if-lauseilla, vaan käytä **try-except**-rakennetta! Muista, että **except**-lausekkeita voi olla useampiakin eri virhetyyppejä varten samassa **try**-lausekkeessa! (ks. materiaalit, **try-except-except**)

   Tehtävän tiedostonimi = ***exercise4_4.py***

   Tyypillinen koodimäärä***: 9-16 riviä*** *(tyhjiä rivejä ja kommentteja ei lasketa)*

5. Tee ohjelma, joka kysyy käyttäjältä onko hän opiskelija vai ei (k / e) sekä kuukauden numeron, johon hän on varaamassa matkalippua. Tee ohjelmaan heti alussa lisäksi Boolean –muuttuja, jonka tehtävänä on osoittaa, onko käyttäjä oikeutettu alennettuun hintaan vai ei (esim. muuttuja nimeltä **special_price**).

   Tarkista ohjelmassa tämän jälkeen ehtolauseita hyödyntäen, onko käyttäjä oikeutettu alennettuun hintaan vai ei. Jos käyttäjä on oikeutettu alennettuun hintaan, muuta **special_price** –muuttuja arvoon ***True***, ja muussa tapauksessa muuta se arvoon ***False***.

   Käyttäjä on oikeutettu alennettuun hintaan, jos hän ilmoittaa olevansa opiskelija (**k**), ja varaus **ei osu** kuukausille 6-8.

Ohjelman koodin tulee noudattaa seuraavanlaista rakennetta (Moodlessa helpommin kopioitava versio):

```
# Ohjelman alkuosa, annetaan special_price:lle alkuarvo,
# voi aloittaa myös arvosta True, jos haluat toimia käänteisellä logiikalla:
special_price = False

# Ohjelman pääosa:
# kysy tässä onko käyttäjä opiskelija vai ei, sekä kuukauden numero (input() jne.)

# Lisää tähän kohtaan kaikki tarvittavat ehtolauseet!
# Ehtolauseiden avulla (vrt. onko opiskelija, onko kesäaika),
# muuta special_price tarvittaessa arvoon True tai False,
# mikäli käyttäjä on tai ei ole kelvollinen alennettuun hintaan

# Tulosta ohjelman lopussa special_price -muuttujan perusteella tieto siitä,
# onko käyttäjä oikeutettu alennukseen
if special_price:
    # tulostetaan ”Alennus myönnetty!”
else:
    # tulostetaan ”Normaali hinta.”
```

Huom: Älä kysy varsinaista hintaa tässä tehtävässä, tarkoitus on vain selvittää kelpoisuus alennukseen. Älä myöskään käytä **exit()** –funktiota tässä tehtävässä!

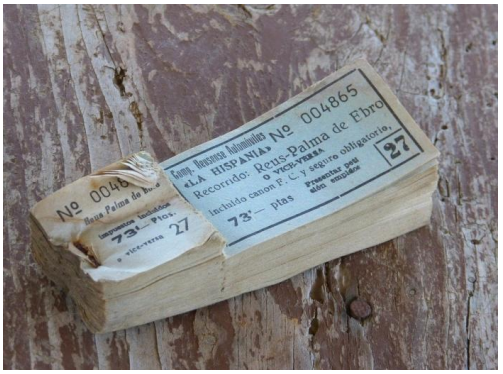
Vinkki: Kuukauden tarkistamisessa kannattaa hyödyntää Pythonista löytyvää kahden arvon vertailua, erillinen esimerkki: **if 0 < number < 10:** jne. (eli onko numero suurempi kuin 0 ja pienempi kuin 10). **Älä testaa erikseen onko kuukausi 6, 7 tai 8**, vaan käytä yllä olevan esimerkin mukaista raja-arvoa ☺ (koska tällainen koodi on huomattavasti paremmin skaalautuva, jos raja-arvot joskus muuttuisivatkin).

Esimerkkejä ohjelman toiminnasta:

```
Oletko opiskelija? (k/e)
k
Mille kuukaudelle matka varataan? (1-12)
4
Alennus myönnetty!
```

```
Oletko opiskelija? (k/e)
k
Mille kuukaudelle matka varataan? (1-12)
7
Normaali hinta.
```

```
Oletko opiskelija? (k/e)
e
Mille kuukaudelle matka varataan? (1-12)
11
Normaali hinta.
```



Tehtävän tiedostonimi = ***exercise4_5.py***

Tyypillinen koodimäärä***: 12-20 riviä*** *(tyhjiä rivejä ja kommentteja ei lasketa)*

LISÄTEHTÄVIÄ:

6. Kysy käyttäjältä tämän hetken ulkolämpötila ja kosteusprosentti. Tee ohjelmaasi myös seuraavat Boolean-muuttujat:
   - **freezing** = False
     heatwave = False
     raining = False
     hailstorm = False

   Jos annettu lämpötila on alle 0 astetta, aseta **freezing**-muuttuja todeksi. Jos kosteusprosentti on yli 80, aseta **raining**-muuttuja todeksi, paitsi jos pakkasta on yli 20 astetta, jolloin asetetaan sen sijaan **hailstorm**-muuttuja todeksi. Jos lämpötila on yli 20 astetta, aseta **heatwave**-muuttuja todeksi.

   Tulosta lopuksi annettu lämpötila. Jos **freezing**–muuttuja on tosi, tulosta ”Pakkasta.”. Jos **raining**–muuttuja on tosi, tulosta ”Sataa.”, ja jos **hailstorm**–muuttuja on tosi, tulosta ”Sataa rakeita!”. Jos **heatwave** –muuttuja on tosi, tulosta ”Helleaalto!”. Jos sataa ja on hellettä, tulosta ”Kosteaa ja tukalaa.”

   ***Huom**: Booleanien tehtävä on usein helpottaa erilaisten toisiinsa liittyvien ehtojen täyttymistä. Ilman Boolean-muuttujia ehtolauseiden ehdot voivat olla joskus melkoisen monimutkaisia!*

   Tehtävän tiedostonimi = ***exercise4_6.py***

   Tyypillinen koodimäärä***: 20-36 riviä*** *(tyhjiä rivejä ja kommentteja ei lasketa)*

7. Tee ohjelma, joka kääntää numerot sanoiksi. Eli jos käyttäjä syöttää ”1465”, ohjelma tulostaa ”yksi neljä kuusi viisi”. Hyväksy maksimissaan viisinumeroiset luvut käyttäjältä.
    - **Lisätehtävä**: Anna käyttäjän syöttää luku myös tekstinä, eli ”yksi kaksi kolme” muuttuisi arvoksi 123. Tässä tapauksessa päättele käyttäjän input() –arvosta syöttikö hän luvun vai tekstiä (ks. tekstifunktiot, esim. isnumeric()),
      eli älä kysy kahdella eri input():lla peräkkäin ensin numeroita ja sitten tekstejä!

Tehtävän tiedostonimi = ***exercise4_7.py***

Tyypillinen koodimäärä***: 24-32 riviä*** *(tyhjiä rivejä ja kommentteja ei lasketa). Lisätehtävän kanssa koodirivejä voi olla jopa 48-60 riviä.*

Tämän tehtävän voi tehdä myös huomattavan paljon pienemmällä koodimäärällä, mutta siihen tarvitaan toistolauseita ja/tai kokoelmia. Käymme toistolauseet ja kokoelmat läpi myöhemmin opintojaksolla!

Huom! Tässä tehtävässä ei oleteta käytettävän toistolauseita tai kokoelmia (saa toki käyttää jos haluaa). Tehtävän voi tehdä täysin ehtolauseiden ja replace()-funktion avulla. ☺