Python-ohjelmointi Harjoitus 10

TAVOITTEET

- Kerrataan harjoituksen 9 keskeisiä sisältöjä.
- Kerrataan listarakenteen perustoiminnot ja opitaan lisää lista-rakenteen hyödyntämistä yhdessä satunnaisuuden kanssa.
- Opitaan työskentelemään merkkijonoilla.
- Opitaan käyttämään merkkijonokirjastoa string.

Matematiikan sisällöt

Tehtävissä harjoitellaan ja syvennetään kirjainlaskennan osaamista.

Ohjelmointiin käytettävä alusta löytyy osoitteesta http://sade-oppiminen.herokuapp.com/

KERTAUSTA HARJOITUKSESTA 9

TEHTÄVÄ 1

Tavoitteena on tehdä ohjelma, jossa kerrataan erilaiset potenssimerkinnät ja math-kirjaston käyttö.

Run

Tee ohjelma, joka laskee ja tulostaa seuraavat laskutoimitukset.

Lasku	**Käytä tätä toimintoa**
9^3	Potenssimerkintä
$5{,}1 \cdot 10^6$	Kymmenpotenssimuoto
7^4	Potenssikomento

Koodaa alla olevat laskutoimitukset haluamallasi toiminnolla.

a) $3^7 \cdot 3^{-5}$

b) $\frac{12^2}{12^{-4}}$

c) $2 \cdot 10^3 + 6 \cdot 10^1 + 10^0 + 7 \cdot 10^{-1}$

KERTAUSTA LISTOISTA

Harjoituksessa 8 tutustuttiin tietovarastoon lista sekä opittiin määrittämään listan pituus, yhdistämään erillisiä listoja yhdeksi, lisäämään listaan alkioita ja korjaamaan listan alkion tietoja.

TEHTÄVÄ 2

Tavoitteena on palauttaa mieleen, miten lista tehdään ja miten listan indeksointi toimii. Lisäksi muistellaan, miten listalta tulostetaan erilaisia osajoukkoja.

Run

Tee ohjelma, jossa listalta tulostetaan erilaisia kokoonpanoja. Yritä vastata kysymyksiin ennen kuin lisäät koodin ohjelmaan ja suoritat sen.

a) Tee lista *numerot*, jossa on luvut 0–10.
b) Mitä tulostuu komennolla print numerot[1:5]?
c) Mitä tulostuu komennolla print numerot[:4]?
d) Mitä tulostuu komennolla print numerot[len(numerot)/2]?
e) Mitä tulostuu komennolla print numerot[len(numerot)–3:]?
f) Mitä tulostuu alla olevalla koodilla?

```
k = 0
while k < len(numerot):
    print numerot[k]
    k = k + 1
```

TEHTÄVÄ 3

Tavoitteena on tehdä ohjelma, jossa luodaan lista ja tulostetaan listan alkioita.

Run

Tee ohjelma kohtien a–f avulla.
a) Lisää ohjelmaan lista *vokaalit* ja sen alkioiksi kaikki vokaalit.
b) Tulosta kaikki listan alkiot.
c) Tulosta listan ensimmäinen alkio.
d) Tulosta listan kolmas alkio.
e) Tulosta listan viimeinen alkio. Voit hyödyntää listan pituutta viimeisen alkion selvittämiseen.
f) Tulosta listan kaksi viimeistä alkiota.

LISÄÄ LISTAN TOIMINNALLISUUKSIA

Listalle tehtäviä toimintoja	
listan kopioiminen	*kopio = alkuperainen[:]* tai *kopio = list(alkuperainen)*
listan järjestäminen aakkos- tai suuruusjärjestykseen	*lista.sort()*
listan kääntäminen	*lista.reverse()*

Esimerkki 1

Tee ohjelma, jossa käsitellään kirjainlaskennan käsitteitä sisältävää listaa. Listalla ovat käsitteet *termi, monomi, kerroin, polynomi* ja *asteluku*.

a) Listan nimi ei voi sisältää ääkkösiä, joten luo lista *kasitteet*.

b) Kopioi lista *kasitteet* uuteen listaan *kopio*.

c) Järjestä lista *kasitteet* aakkosjärjestykseen.

d) Käännä listan *kasitteet* järjestys.

e) Tutki, onko käsite *kerroin* listalla *kasitteet*. Testaa samaa vielä käsitteellä *vakiotermi*.

Ratkaisu

Jotta listalle tehdyt toimet nähdään, tulostetaan käsiteltävänä oleva lista jokaisessa vaiheessa. Tuloste näkyy koodin alla harmaalla pohjalla.

```
# a) Luodaan lista kasitteet.
kasitteet = ['termi', 'monomi', 'kerroin', 'polynomi', 'asteluku']
print kasitteet
```

```
['termi', 'monomi', 'kerroin', 'polynomi', 'asteluku']
```

```
# b) Kopioidaan lista kasitteet uuteen listaan kopio.
kopio = kasitteet[ : ]
print kopio
```

```
['termi', 'monomi', 'kerroin', 'polynomi', 'asteluku']
```

```
# c) Järjestetään lista kasitteet aakkosjärjestykseen. Järjestäminen muokkaa
alkuperäistä listaa.
kasitteet.sort()
print kasitteet
```

```
['asteluku', 'kerroin', 'monomi', 'polynomi', 'termi']
```

```
# d) Käännetään listan kasitteet järjestys.
kasitteet.reverse()
print kasitteet
```

```
['termi', 'polynomi', 'monomi', 'kerroin', 'asteluku']
```

```
# e) Tutkitaan, ovatko tietyt käsitteet listalla.
# Määritellään tutkittava käsite muuttujaksi etsi.

etsi = "kerroin"
k = 0
while k < len(kasitteet):
  if kasitteet[k] == etsi:
    print etsi + u" löytyy listalta."
  k = k + 1
```

```
kerroin löytyy listalta.
```

TEHTÄVÄ 4

Tavoitteena on tehdä ohjelma, jossa harjoitellaan listan toimintojen käyttämistä.

Run

Tee ohjelma alla olevien kohtien mukaan. Jotta listalle tehtävät toimet saa näkyviin, lista pitää tulostaa jokaisen vaiheen jälkeen.

- Luo lista *oppiaine* ja lisää siihen oppiaineita.
- Järjestä lista aakkosjärjestykseen.
- Käännä listan alkiot käänteiseen järjestykseen.

TEHTÄVÄ 5

Tavoitteena on tehdä ohjelma, jossa harjoitellaan listan toiminnallisuuksia sekä kahden listan yhdistämistä.

Run

Tee ohjelma alla olevien kohtien mukaan.

- Luo lista *etunimi*, ja lisää siihen etunimiä.
- Luo lista *sukunimi*, ja lisää siihen sukunimiä. Lisää listaan yhtä monta nimeä kuin on *etunimi*-listassa.
- Luo uusi lista *nimi*, johon tallennetaan etunimestä ja sukunimestä yhdistetty nimi. Listan alkiossa sukunimi on ensin. Hyödynnä silmukkarakennetta listan alkioiden lisäämiseen.
- Järjestä lista *nimi* aakkosjärjestykseen.
- Tulosta lista *nimi* alkio kerrallaan. Käytä tulostamisessa silmukkarakennetta.

MERKKIJONOT

Merkkijono on yksittäisistä merkeistä koostuva lista. Merkkijonoa käsitellään kuten listaa eli käytetään samoja komentoja ja tarkentimia.

Merkkijonossa jokaisella merkillä oma paikkansa eli indeksinsä, kuten listassakin. Ensimmäisen merkin indeksi on nolla, toisen 1 ja niin edelleen. Myös välilyönti on oma merkkinsä eli se saa oman indeksin.

MERKKIJONO

merkkijono = ”merkit jonossa”

Jokaisella merkillä on oma indeksi eli paikka.

m	e	r	k	i	t		j	o	n	o	s	s	a
0	1	2	3	4	5	6	7	8	9	10	11	12	13

Esimerkki 2

Tee ohjelma, jossa käyttäjältä kysytään kaksi sanaa. Nämä kaksi sanaa muodostavat merkkijonon. Tämän jälkeen merkkijonosta tulostetaan

- koko merkkijono
- ensimmäinen merkki
- viimeinen merkki
- keskimmäinen merkki
- ensimmäinen sana
- merkkijono käännettynä.

Ratkaisu

```
# Pyydetään käyttäjältä merkkijono eli kaksi sanaa.
# Käyttäjän antamat sanat talletetaan muuttujaan merkkijono.

merkkijono = input('Anna merkkijonoksi kaksi sanaa.')

print merkkijono
# Merkkijonon ensimmäisen merkin indeksi on 0.
print merkkijono[0]

# Merkkijonon viimeinen merkki saadaan selvittämällä ensin merkkijonon pituus.
Viimeisen merkin indeksi on jonon pituus - 1.
print merkkijono[len(merkkijono)-1]

# Merkkijonon keskimmäisen merkin indeksi saadaan, kun merkkijonon pituus jaetaan
kahdella.
print merkkijono[len(merkkijono)/2]

# Merkkijonon ensimmäinen sana saadaan etsimällä välilyönnin indeksi.
Merkkijonosta tulostetaan alusta välilyönnin indeksiin asti.

a = merkkijono.index(" ")
print merkkijono[ : a]
```

```
# Määritetään uusi muuttuja käännetylle merkkijonolle. Alustetaan
# merkkijonoKaannettyna tyhjäksi.

# Alkuperäinen merkkijono saadaan käännettyä, kun lähdetään tarkastelemaan sitä
# viimeisestä merkistä. Silmukkarakenteella käydään läpi alkuperäinen merkkijono
# ja lisätään merkkejä käännettyyn merkkijonoon.

merkkijonoKaannettyna = ("")
k = len(merkkijono) - 1
while k > -1:
    merkkijonoKaannettyna += merkkijono[k]
    k = k - 1

print merkkijonoKaannettyna
```

Ohjelman testaamiseksi siihen syötettiin merkkijonoksi ”luminen aamu”. Ohjelman tuloste on seuraava:

```
luminen aamu
l
u
n
luminen
umaa nenimul
```

MERKKIJONOJEN KOMENTOJA	
merkkijonon pituus	len(*merkkijono*)
merkin etsiminen merkkijonosta	merkkijono.find(*”etsittävä merkki/merkit”*)
merkin korvaaminen toisella merkillä	merkkijono.replace(*”korvattava merkki”, ”korvaava merkki”*)
merkkijonon jakaminen osiin (osat muodostavat listan)	osat = merkkijono.split(*”merkki, jonka kohdalta jaetaan”*)
merkin lisääminen merkkijonon loppuun	merkkijono = merkkijono + *”lisättävä merkki”*
merkkijonojen yhdistäminen	uusijono = *merkkijono1* + *merkkijono2*

TEHTÄVÄ 6

Tavoitteena on tehdä ohjelma, jolla harjoitellaan merkkijonoille käytettäviä komentoja ja toimintoja.

Run

Tee ohjelma alla olevien kohtien mukaan.

- Pyydä käyttäjältä jokin lause.
- Määritä lauseen pituus.
- Muuta lauseen lopettava piste välilyönniksi.
- Jaa lause välilyöntien kohdalta sanoiksi.
- Tulosta sanat käyttäen silmukkarakennetta. Silmukkaa toistetaan sanojen verran. (Vihje: selvitä sanojen lukumäärä eli listan osien pituus.)

TEHTÄVÄ 7

Tavoitteena on harjoitella merkkijonoille käytettäviä toimintoja.

Run

Tee ohjelma, jossa käyttäjältä pyydetään tietyn termin kanssa samanmuotoisia termejä. Ohjelma antaa palautteen, onko käyttäjän antama termi samanmuotoinen vai ei. Tee ohjelma, joka vastaa alla olevaa suunnitelmaa.

Koska Python ei tue yläindeksiä, eksponentteja sisältävä kirjainosa kirjoitetaan ohjelmaan muodossa 7*x*3. (Matematiikassa kirjoitusasu on $7x^3$.)

- Määritä, minkä termin kanssa samanmuotoisia termejä halutaan kysyä. Käyttäjän antama vastaus tallennetaan muuttujaan *samanmuotoinen*.
- Lisää käyttäjän antaman merkkijonon perään välilyönti.
  (Vihje: *merkkijono* = *merkkijono* + ” ”)
- Erota annetusta termistä kirjainosa. Sitä varten täytyy etsiä indeksi, josta haluttu kirjainosa löytyy.
- Määritä, millaista kirjainosaa etsitään. Välilyönti lopussa auttaa erottamaan eksponentteja sisältävät termit.
- Vertaa käyttäjän antamaa kirjainosaa etsittyyn kirjainosaan. Mikäli ne ovat samat, käyttäjä saa tiedon, että hän antoi samanmuotoisen termin. Muutoin kerrotaan, ettei annettu termi ole samanmuotoinen.

TEHTÄVÄ 8

Tavoitteena on tehdä ohjelma, jolla tutkitaan, onko annettu merkkijono palindromi eli sama molemmista suunnista luettaessa. Esim. ANNA.

Run

Tee ohjelma alla olevien kohtien mukaan.

- Pyydä käyttäjältä merkkijono.
- Poista annetusta merkkijonosta välilyönnit lopullisen vertailun helpottamiseksi.
- Tee uusi muuttujan, johon tallennat merkkijonon käännettynä.
- Vertaa alkuperäistä ja käännettyä merkkijonoa. Jos ne ovat samat, on käyttäjän antama merkkijono palindromi.

Esimerkki 3

Tee ohjelma, jossa otetaan käyttöön merkkijonokirjasto *string* ja arvotaan pienistä kirjaimista viiden merkin mittaisia ”sanoja”.

STRING-kirjasto	
kirjaimet	*string.letters*
pienet kirjaimet	*string.ascii_lowercase*
isot kirjaimet	*string.ascii_uppercase*
numerot	*string.digits*

Ratkaisu

```
# Lisätään kirjastot string ja random.
import string
import random

# Määritetään, että sana on aluksi tyhjä.
sana = ""

# Silmukkarakenteella arvotaan 5 merkkiä, joista sana muodostuu. Jokaisella
kierroksella arvotaan yksi merkki pienien kirjaimien joukosta ja lisätään sitten
arvottu merkki sanaan.
for k in range(5):
    a = random.choice(string.ascii_lowercase)
    sana = sana + a
# Tulostetaan muodostunut "sana".
print sana
```

Arvotut sanat ovat vain joukko merkkejä, eikä niistä välttämättä muodostu mitään järkevää.

TEHTÄVÄ 9

Tavoitteena on tehdä ohjelma, jolla harjoitellaan string- ja random-kirjastojen käyttämistä yhdessä.

Run

Tee ohjelma, joka arpoo kymmenen merkkiä pitkän salasanan. Salasanassa saa olla isoja kirjaimia, pieniä kirjaimia ja numeroita.

Vihje: arvo merkit joukosta, jonka muodostavat kirjaimet ja numerot.

TEHTÄVÄ 10

Tavoitteena on tehdä ohjelma, jossa tutkitaan, ovatko kaksi sanaa anagrammeja eli ne muodostuvat samoista kirjaimista.

Run

Tee ohjelma, jossa käyttäjä antaa kaksi sanaa. Sanat ovat anagrammeja, mikäli niissä esiintyy samoja kirjaimia ja kirjaimien lukumäärät ovat samat molemmissa sanoissa.