



Työpöytä ► Omat opintojaksoni ► 1. Ohjelman perusrakenne, datatyypit, operaattorit... ►
Oppimistehtävä 1

This Moodle environment is for archive use only. All the active courses at the beginning of the year 2020 are held in the [Learn](#) and [Open Learn](#) environment.

Oppimistehtävä 1

Tulostus- ja syötteet, muuttujat, operaattorit ja datatyypit

Lataa tehtävään tarvittava zip-tiedosto ja pura tiedosto levyillesi kansioksi. Kansiossa oleva ot1.js on tehtävätiedosto, johon sinun tulee tehdä kaikki tämän tehtäväsarjan tehtävät! Kansioista löytyy myös ot1.html -tiedosto, jonka voit avata verkkoselaimeesi. Suosittelen Google Chromea sen erinomaisten kehittäjän työkalujen takia. Muut kansiossa olevat tiedostot ovat tehtävään tehtyjä ohjaus- ja tyyli-tiedostoja, joita sinun ei tule muuttaa. Voit kuitenkin halutessasi tutkia esim. ot1.html:n ja js/app.js:n lähdekoodeja, jos sinua kiinnostaa miten koko tehtävä on konepellin alla ohjelmoitu.

Jos sinulla tulee ongelmia ot1.html -verkkosivun avaamisessa tekemiesi muokkauksien jälkeen, voit yrittää ratkaista vikaa kehittäjän työkalujen avulla (<https://javascript.info/debugging-chrome>). Voit esimerkiksi tulostaa konsoliin haluamiasi tietoja console.log -funktiolla, jonka voit asettaa tarvittaessa mihin tahansa koodiisi.

Aloitetaan!

a) Muuttujat

Muuttujia käytetään JavaScript ohjelmoinnissa datan ylläpitämiseen ja manipulointiin ajonaikaisesti.

Moderni tapa muuttujan luomiseen on käyttää 'let' -syntaksia:

```
let muuttujanNimi = 'Muuttujan tekstisisältö';
```

Tämä lauseke siis luo muuttujan nimeltä "muuttujanNimi" ja asettaa sen arvoksi "Muuttujan tekstisisältö".

Toinen, vanhempi tapa luoda muuttujia on käyttää 'var' -syntaksia, kuten alla:

```
var muuttujanNimi = 'Muuttujan tekstisisältö'
```

Tätä tapaa ei kuitenkaan suositella, sillä se voi aiheuttaa ongelmia laajoissa sovelluksissa, joten käytä tällä kurssilla ainoastaan 'let' -syntaksia.

Muuttujan arvoa voidaan muuttaa seuraavalla syntaksilla:

```
muuttujanNimi = 'Muuttujan uusi tekstisisältö'
```

Huomaa, että tämä lause yliajaa edellä annetun arvon muuttujan uudella arvolla.

Muokkaa seuraavaksi tehtäväohjelman tiedostoa ot1.js, ja luo sinne muuttuja nimeltä "tervehdys_a" ja alusta se arvolla "Heippa maailma!".

Voit tarkistaa ohjelman toimivuuden avaamalla ot1.html -tiedoston verkkoselaimellasi.

b) Vakiot

Muuttujien arvoa voidaan muuttaa ohjelman ajon aikana. Jos kuitenkin tiedetään että kyseisen arvon ei pitäisi muuttua, voidaan tieto varastoida vakioon const -syntaksilla:

```
const vakionNimi = 'Tätä tekstiä ei voi muuttaa';
```

Tätä pidetään hyvänä käytäntönä, sillä koodia lukiessa toinen ohjelmoija voi luottaa heti vakion nähdessään, että sen arvo ei tule muuttumaan.

Muokkaa seuraavaksi tehtäväohjelman tiedostoa ot1.js, ja luo sinne vakio nimeltä "tervehdys_b" ja alusta se arvolla "Heippa vakiomaailma!"

c) Merkkijonot ja string-liitokset

Merkkijonot, eli stringit, ovat yksi JavaScriptin datatyypeistä. Niillä kuvataan tekstimuotoista dataa, ne merkitään heittomerkkisyntaksilla, kuten edellisessä tehtävässä huomasimme. Niitä voidaan liittää toisiinsa "+" -operaattorilla, esimerkiksi:

```
const etunimi = 'Aatu';  
const sukunimi = 'Auvinen';  
const kokonimi = etunimi + ' ' + sukunimi;
```

Huomaa, että esimerkin kokonimi-muuttujaan yhdistettiin kaikkiaan kolme merkkijonoa, joista yksi ole pelkkä tyhjä välilyönti, jotta arvoksi saadaan siististi luettava 'Aatu Auvinen'.

Muokkaa seuraavaksi tehtäväohjelman tiedostoa ot1.js, ja luo tervehdysteksti käyttäen annettuja muuttujia niin että tervehtijä ja tervehditty esiintyvät tervehdystekstissä. Jos esimerkiksi tervetijä on "Minä" ja tervehdittävä on "Maailma", tulee tervehdystekstiksi muodostua "Heippa Maailma! Terveisin, Minä"

d) Boolean-arvot ja negaatio

Yksi JavaScriptin datatyypeistä on boolean. Booleanilla voi olla kaksi arvoa, true ja false, eli tosi ja epätosi. Boolean-arvoja käytetään monenlaisissa vertailulausekkeissa ja kontrollirakenteissa, mutta niistä lisää seuraavissa harjoituksissa. Tässä vaiheessa on oleellista tietää, että tällainen datatyyppi löytyy, ja että boolean-arvot alustetaan seuraavalla syntaksilla:

```
const tosi = true;  
const epatosi = false;
```

Booleanin alustus siis muistuttaa merkkijonon alustusta, mutta true- ja false-teksteihin ei lisätä heittomerkkejä. Mikäli arvot alustettaisiin esim. 'true' -arvolla, ei kyseessä olisi enää boolean. Booleanin kanssa voidaan käyttää negaatio-operaattoria, joka antaa vastakkaisen

arvon, kun mikä negaation lähtöarvo on. Negaation syntaksissa käytetään huutomerkkiä, kuten alla:

```
const epatosi = !true;
```

Muokkaa seuraavaksi tehtäväohjelman tiedostoa ot1.js, ja määrittele haeTosi-funktion arvo-vakio, niin että se saa arvokseen boolean "true", ja haeVastakohta-funktion arvo-vakio niin, että se on annetun muuttujan vastakohta

e) Numeeriset arvot ja matemaattiset operaattorit

Numerot ovat yksi JavaScriptin datatyypeistä. Niitä käytetään matemaattiseen laskentaan, ja ne myös vie vähemmän muistia kuin merkkijonot, joten pääsääntöisesti puhtaasti numeerisia arvoja tulisi käsitellä numero-datatyypillä (Number). Numero-arvon alustuksella on seuraavanlainen syntaksi:

```
const kokonaisluku = 10;  
const liukuluku = 123.456;  
const negatiivinen = -10
```

Numerot voivat siis olla kokonaislukuja, desimaalilukuja tai negatiivisia lukuja. Numeroita voidaan käyttää laskennassa mm. operaattorien "+", "-", "/", "*" ja "%" kanssa, syntaksilla:

```
const summa = 100 + 5;
```

Jolloin summan arvoksi tulee 105.

Muokkaa seuraavaksi tehtäväohjelman tiedostoa ot1.js, ja määrittele oppimistehtävään 1 E merkityt arvot niin, että annettuun numeroon lisätään yksi, vähennetään yksi tai se nollataan

f) Matemaattisten operaattorien soveltaminen

Täydennä ot1.js:n oppimistehtävä 1 F:n koodiin annetut kohdat niin, että annetut luvut summataan, vähennetään, kerrotaan tai jaetaan toisillaan

g) Tyypimuunnokset

JavaScript on dynaamisesti tyyplitetty kieli. Tämä tarkoittaa sitä, että muuttujan tyyppi voidaan muuttaa toiseen tyyppiin ajonaikaisesti, ja usein tämä tehdään automaattisesti. Katsotaan esimerkkiä:

```
const artistinimi = 'Jussi' + 60;
```

Tässä tapauksessa numeerinen arvo 60 muutetaan merkkijonoksi, jotta merkkijonon liitosoperaatio olisi mahdollinen, ja artistinimen arvoksi tulee 'Jussi60'. Joskus automaattiset tyyppimuutokset eivät kuitenkaan riitä, vaan ohjelmoijan täytyy tehdä tyyppimuunnokset itse. Merkkijonoon muuttamiseksi JavaScriptissä käytetään .toString -funktiota, joka toimii seuraavasti:

```
const vuosisata = 19;
const vuosiluku = 17;
const itsenaisyys = vuosisata.toString() + vuosiluku.toString();
```

Muita toimivia esimerkkejä olisi:

```
const itsenaisyys = 19.toString() + 17.toString()
const itsenaisyys = '' + 19 + 17;
```

Jälkimmäisin esimerkki demonstroii, että toString()-funktiota ei tarvitse välttämättä kutsua itse. Jos aluksi yritetään tehdä string-operaatio, jossa tyhjään merkkijonoon yhdistetään numero 19, muutetaan numero merkkijonoksi, millä operaatio toteutetaan. Seuraavaan operaatioon (+ 17) siirryttyä yritetään jälleen liittää numeroa merkkijonoon, jolloin sama automaattinen tyyppimuutos toteutuu. Sitä vastoin, lauseke: 19 + 17 + " palauttaisikin merkkijonon arvolta '36', sillä numeeristen arvojen ynnäys on tehty ennen kuin merkkijonon liitosoperaatioon on päästy.

Merkkijonosta numeroksi muuntaminen sen sijaan onnistuu funktiolla Number() seuraavasti:


```
const luku1 = '20';
const luku2 = '35';
const summa = Number(luku1) + Number(luku2);
```

Summaksi tulee haluttu 55. Jos Number-funktiota ei olisi tässä käytetty, olisi summan arvoksi tullut merkkijono '2035'.

Muokkaa seuraavaksi tehtäväohjelman tiedostoa ot1.js, ja muuta oppimistehtävä 1 G:n muuttujien tyypit niin, että operaatioista syntyy oikea tulos


Kun olet tehnyt kaikki tehtävät, pakkaa koko tehtäväkansio zip-tiedostoon, ja palauta Moodleen allaolevan palautuslinkin kautta.

Palautuksen tila

Suorituskerran numero	Tämä on suorituskerta 1
Palautuksen tila	Lähetetty arvioitavaksi
Arvioinnin tila	Arvioitu
Viimeksi muokattu	tiistai, 1 lokakuu 2019, 15:32
Tiedostojen palautus	 ot1_tehtavat_asko.rar
Palautuksen lisätiedot	► Kommentit (0)

Muokkaa palautusta

Palaute

Arviointi	hyväksytty
Arviointipäivä	maanantai, 6 tammikuu 2020, 20:03
Arvioija	 Teemu Salminen
Palaute	Muuten OK, mutta tervehtijä ja tervehdittävä ovat väärin päin tehtävässä C.

NAVIGOINTI



Työpöytä

[Sivuston etusivu](#)

[Omat opintojaksoni](#)

[Opintojaksokategoriat](#)

[Moodle-alustan tilaus \(Opettajille\)](#)

ASETUKSET



[Opintojakson ylläpito](#)



