# ISETL-Harjoitus 1, 1999. ISETL ja jaollisuus

## Tehtävä 1. Laskentoa, pötköjä ja ohjelmointia

Kirjoita seuraavat rivit, ja tutki mitä ne tarkoittavat.

```
>  7 + 18;
>  13*(344);
>  5**2;
>  6 = 2*3;
>  5 >= 2 * 3;
>  150
>>  +23-365
>>  *2
>>  ;
>  x := -23.5;
>  x;
>  x := x + 20;
>  x;
>
>  23/34;
>  !rational on
>  3/4 + 5/6;
>  !rational off
>  3/4 + 5/6;
>
>  p := [3, -2]; q := [1, 4.5]; r := [0.5, -2, -3];
>  p; q; r;
>  p(1); p(2); q(2); r(4);
>  pituus := 0;
>  for i in [1..3] do
>>  pituus := pituus + r(i)**2;
>>  end;
>  pituus := sqrt(pituus);
>  pituus;
```

Varmistakaa, että kaikki ryhmän jäsenet osaavat vastata seuraaviin kysymyksiin:

a) Mitä eroa on ISETL-kehotteella > ja kehotteella >>?
b) Mitä `OM` tarkoittaa (kun kirjoitit `r(4)`)?
c) Mikä merkitys on merkillä `:=` ?
d) Mitä viimeisessä koodinpätkässä tapahtuu?

Tallentakaa kunkin tehtävän ratkaisu tekeminenne virheineen levykkeelle sopivalla nimellä, esimerkiksi `1HarjT1.ise` (niin kuin 1. Harjoitus, Tehtävä 1). Yleensä jokainen tehtävä kannattaa tallentaa erikseen, jotta ne olisivat helposti myöhemmin löydettävissä.
Ottakaa uusi tiedosto sulkemalla edellinen valikosta **File - Close**.

**Tehtävä 2. Totuus, joukot ja tuplet. Permutaatiot ja jakaminen**

Ennen kuin kirjoitat seuraavat koodit, arvaa, mitä ne tuottavat ja kirjoita arvauksesi ylös.

```
>  Maksim := (max(2,3) = 2);
>  maksim;
>  Maksim;
>  Maksim or (5 > 4) ;
>  sqrt(7) = 7**0.5;
>  (2 /= 3);              $ merkki /= tarkoittaa erisuuruus-merkkiä
>  (2 /= 3) and ((5/3) > 1);
>  abs(2-4) < 1; $ 'abs' kuten absolute value; itseisarvo
>  K := {1,2,3,4}; H := {2,4,6}; G := {};
>  K inter H;             $ 'inter' tulee sanasta intersection
>  K union H;
>  #K; #H; #G;
>  G inter H;
>  Z20 := {0..19}; Z20;
>  kolmeZ20 := {0,3..19}; kolmeZ20;
>  kolmeZ20 subset Z20;   $ subset = osajoukko
>  1 in K; 0 in K;
>  1 in K-{1};
>  K-{1};
>  A := [1..7]; A; A; A;
>  B := {1..7}; B; B; B;
```

Mikä ero on `A`:lla ja `B`:llä? (`A` on ISETLissä datatyyppiä TUPLE).

```
>  p := [3,2,1]; q := [3,1,2]; r := [p(q(i)) | i in [1..3]];
>  r;
>  S3 := { [a,b,c] | a,b,c in [1..3] : #{a,b,c} = 3 };  S3;
```

Joukon `S3` alkioita kutsutaan permutaatioiksi (tavoiksi järjestää kolmen alkion joukko), ja `r` on permutaatioiden `p` ja `q` yhdiste.
Tutki seuraavaksi ISETL-komentoja `div` ja `mod` erilaisilla luvuilla. Mitä nämä funktiot tekevät?

```
>  6 div 3; 7 div 3; 8 div 3; ... (jatka samaan tapaan)
>  6 mod 3; 7 mod 3; 8 mod 3; ...
```

Mikä yhdistää lukuja $m$, $n$, $m$ div $n$ ja $m$ mod $n$ ?

**Tehtävä 3. Funktiot ISETLissä, tekijän testaus**

Kirjoita ensin seuraavat koodinpätkät ja tutki, mitä ne tarkoittavat:

```
>  g := func(x);
>>  return 2*x - x**2;
>>  end func;
>  g(2); g(0);
```

Seuraavassa funktiossa esiintyvä sana 'integer' on suomeksi *kokonaisluku*.

```
>  a10 := func(x,y);
>>  if (is_integer(x) and is_integer(y)) then
>>    return (x+y) mod 10
>>    else return ''Eivat olleet kokonaislukuja''
>>  end if;
>>  end;
>  a10(3,4); a10(1,11); ...
```

Kirjoita sitten oma funktio `onko_tekija`, jonka syötteenä on kaksi lukua $x$ ja $y$, ja joka palauttaa totuusarvon `true`, mikäli $x$ on $y$:n tekijä, ja muutoin palauttaa `false`. Aloita funktiosi kirjoittaminen rivillä

```
>  onko_tekija := func(x,y);
```

Muista, että funktiossa tulee olla `return`-lauseke!

**Tehtävä 4. Jakoyhtälö**

Kirjoita funktio `jakoyhtalo`, jonka syötteenä on kaksi lukua $a$ ja $b$. Tarkista, ettei $b$ ole nolla, mutta että $a$ ja $b$ ovat kokonaislukuja; tämän tekee komento `is_integer` (ks. yllä). Funktion tulee palauttaa lukupari (ts. TUPLE-tyyppinen kokonaisuus). Lukuparin ensimmäinen luku on luvun $\frac{a}{b}$ kokonaisosa ja toinen luku on jakojäännös. Testaa sitten funktiotasi erilaisilla lukupareilla.
Mikä matemaattinen yhtälö voidaan kirjoittaa lukujen $a$, $b$, $q$ ja $r$ välille, jos $q$ on kokonaisosa (eli osamäärä) ja $r$ on jakojäännös luvusta $\frac{a}{b}$? Millaisia arvoja $r$ voi saada?

**Tehtävä 5. Joukkoja ja tupleja**

**a)** Kirjoita sanalliset selitykset kustakin seuraavasta ISETL-ohjelmalle annettavasta rivistä:

```
>  Z20 : = { a mod 20 | a in [-30..50] };
>  H := { g : g in Z20 | even(g) };
>  K := { (5*g) mod 20 | g in Z20 };
>  L := [ [q,r] | q, r in [-50..50] : 45 = 6*q + r ];
>  #(Z20); #H, #K;
>  s := [ i | i in [1..20] : even(i) ];
>  t := { i | i in [1..20] : even(i) };
>  s; t; s; t;
```

**b)** Muodosta joukko, jonka alkioina ovat kaikkien parillisten välillä $[1, 50]$ olevien kokonaislukujen neliöt.
**c)** Muodosta kaikkien niiden välillä $[1, 100]$ olevien kokonaislukujen järjestetty jono (eli TUPLE), joiden jakojäännös jaettaessa luvulla 4 on 3.

### Joukot ja tuplet

ISETLissä voidaan muodostaa äärellisiä joukkoja, eli sellaisia joukkoja, joissa on vain äärellinen määrä alkioita. Aivan kuten matematiikassa, ISETLissä joukon erottaa $\{\}$-sulkeista. Sillä, missä järjestyksessä joukon alkioita luetellaan, ei ole merkitystä. Siis joukot $\{1, 3, 4\}$ ja $\{4, 1, 3\}$ ovat samoja.
Tuple ISETLissä on järjestetty äärellinen jono. Järjestyksellä on siis väliä: tämän voit testata esimerkiksi kirjoittamalla ISETLissä `[1,3,2] = [2,3,1];` Ohjelma palauttaa `false`. Tuplen alkioiden ei tarvitse olla kokonaislukuja; esimerkiksi
`[3,"kolme",2+1]` ja `[x,4,[3,"OM"]]`
ovat tupleja. Suomen kielessä ei ole mitään hyvää nimitystä TUPLElle. Sitä voisi kutsua alkioiden lukumäärän mukaan 2-jonoksi, 3-jonoksi, jne. tai vektoriksi.
Joukkoja voidaan muodostaa käyttämällä syntaksia
`{(joukon alkion muoto) : (ehto, jonka joukon alkion tulee toteuttaa)}`
aivan samaan tapaan kuin matematiikassakin. Niinpä esimerkiksi joukko $\{ k + 2 \mid k \in \{1, 3, 4\}\}$ kirjoitetaan ISETLiin joukkona `{k+2 | k in {1,3,4}}` eli hyvin samalla tavalla kuin matematiikassakin. Joukon alkion muotoa ja ehtoja erottaa joko kaksoispiste ( : ) tai pystyviiva ( | ).
Tupleja voi muodostaa samalla tavalla. Lisäksi ISETLissä ovat käytössä komennot

`{a..b}, [a..b], {a,b..c}` ja `[a,b..c]`

### Tehtävä 6. Suurin yhteinen tekijä SYT

Tee funktio, jonka syötteenä on kaksi lukua $m$ ja $n$, ja joka palauttaa näiden lukujen suurimman yhteisen tekijän. Käytä hyväksesi `onko_tekija`- funktiota. Aseta aluksi suurimmaksi tekijäksi $s := 1$. Käy läpi kaikki kokonaisluvut väliltä $[1, m]$ ja jokaisen kohdalla tarkista, onko luku molempien tekijä. Mikäli luku on molempien tekijä, aseta $s$:n arvoksi tämä luku. Koska tässä ohjelmassa käytetään muuttujaa $s$, joka ei ole funktion syöte, kutsutaan sitä lokaaliksi muuttujaksi. Funktion määrittelyn alkuun kirjoitetaan silloin `local s;`
Ohjelma alkaa siis tällä tavoin:

```
>  syt := func(m,n);
>>  local s;
>>  s := 1;
>>  for i in [1..m] do
....
```