Avaldised

Avaldised *(expression)* moodustavad programmide koostamisel olulise osa. Avaldis võib koosneda vaid ühest väärtusest (konstandist või muutujast), kuid enamasti sisaldab ta ka tehtemärke ja/või funktsioonide väljakutseid. Vastavalt avaldises kasutatavatele ja tulemuseks tekkivale andmetüüpidele jaotatakse avaldisi:

- aritmeetikaavaldisteks
- loogikaavaldisteks
- stringavaldisteks

Avaldistes osalevad operandid ja operaatorid. **Operaatorid** on need, mis opereerivad, st enamasti tehtemärgid. **Operandid** on väärtused ja nendega opereeritakse.

Avaldised võivad olla erinevates kohtades, näiteks:

- Omistuslause paremal poolel – kõige tavalisem olukord. Leitakse (uus) väärtus ja omistatakse see muutujale.
- Väljatrükilauses (print()) – leitud väärtus trükitakse kohe välja.
- Funktsiooni argumendi kohal – leitud väärtus läheb sisendparameetrina funktsiooni.
- Loogikaavaldise koosseisus – leitud väärtus osaleb võrdlustehtes.

Aritmeetikaavadised

Aritmeetikaavaldise tulemuseks on arvuline väärtus (täisarv *(integer)* või ujukomaarv *(floating point number)*). Lihtsamalt öeldes on aritmeetikaavaldise näol tegemist matemaatikast tuttava arvutuseeskirjaga. Aritmeetikaavaldised moodustatakse:

- Konstantidest (arv, näiteks 3 või 2.5).
- Muutujatest (muutujale on eelnevalt antud arvuline väärtus)
- Funktsiooni väljakutsetest (muuhulgas on vaja leida ruutjuurt: `sqrt(arv)`)
- Aritmeetikaoperaatoritest ehk tehtemärkidest `(Pythonis: + - * / // % **)`

Jagamistehte / tulemuseks on ajati ujukomaarv (*float*). Ka siis, kui operandid on täisarvud, mis jaguvad (st 6/2 on 3.0, mitte 3).

Astendamise märk on **, mis on erinev tavapärasest (`2**6` tähendab 2^6). NB! See ei ole ruutu tõstmine, aste tuleb alati lisada peale märki.

Täisarvuline jagamine `//` - nn *floor divison* - leitakse jagatise täisosa nii täisarvude kui ka ujukomaarvude puhul.

Jääk % - täisarvulisel jagamisel tekkiva jäägi leidmine

Aritmeetikaavaldised moodustavad küllaltki tähtsa osa kõigis programmides, eriti nende arvutuslikust osast.

Näit: `arv1 + 2 - sqrt(arv2)` on aritmeetikaavadis, mis koosneb ühest muutujast, konstandist ja funktsioonivälјakutsest ning kahest tehtemärgist.

Tehete järjekord (nn prioriteedid) on sarnane matemaatikast tuttavale. Kõigepealt astendatakse, siis korrutatakse ja jagatakse ning lõpuks liidetakse ja lahutatakse. Samasse kategooriasse kuuluvad tehted tehakse vasakult paremale. Arvestada võib ka sellega, et funktsioon on avaldises nagu üks üksus ja selle väljakutse tehakse kindlasti enne, ku tema väärtusega arvutama hakatakse. Seega fun ktsiooni väljakutsele ei ole vaja eraldi sulge ümber kirjutada. Kui tehete loomulikku järjekorda on vaja muuta, siis sobivad selleks ümarsulud (). Ümarsulge võib olla avaldises rohkem kui üks paar. Lisaks võib sulge kasutada ka avaldise loetavuse tõstmiseks inimese jaoks ning juhul, kui kahtled, kas arvutamine ikka toimub vajalikus järjekorras.

Eelmises avaldise näites leitakse kõigepealt `arv2`-st ruutjuur (kasutades selleks standardfunktsiooni `sqrt()`, seejärel liidetakse muutuja `arv1` ja arvkonstant `2` ning lõpuks lahutatakse juba eelnevalt leitud ruutjuur.

Loogikaavaldised

Loogikaavaldistel võib olla tulemuseks kaks võimalikku väärtust:

tõene `true 1`

väär `false 0`

Näiteks avaldis `3<5` on tõene, aga avaldis `3>5` on väär. Pythonil on konstandid `True` ja `False` tõese ja väära väärtuse tähistamiseks. NB! Need kirjutatakse suure algustähega!

Loogikaavaldised võivad koosneda:

- aritmeetika- või stringavaldistest (mis esindavad väärtust)
- võrdlusoperaatoritest ehk võrdlusmärkidest `< <= > >= == !=`
- loogikatehte märkidest (`and or not`)

`and` – konjunktsioon ehk loogiline ja

`or` – disjunktsioon ehk loogiline või

`not` – loogiline eitus

Võrrelda saab nii arve kui ka stringe. Järgnevusseost (ehk suhteid suurem/väiksem) lubab Python 3.x määrata vaid sama tüüpi andmetele. Seega me võime küsida, kas `3 < 2` või `"kass" > "koer"`. Kuid mitte nii: `2 > "kass"`. See on ka täiesti mõistlik. Viimasel juhul tekib erind (täitmisaegne viga) *Type error*. Tekstide puhul on võrdlemise aluseks kooditabelis peituvad tähtede ja teiste märkide koodid. Teatud mööndusega võib väita, et stringide puhul vastab järjestus tähestikule – see mis tähestikus eespool, on väiksem. See tähendab, et tõesed *(True)* on väited:

`"elevant" < "sääsk"`

`"koer" > "kass"`

`"11" < "3"`, kuid `11 > 3`

Märkidega == ja `!=` võib Pythonis võrrelda ka erinevat tüüpi andmeid (`2 != "kass"`). Iseküsimus, millist infot selline võrdlemine annab.

Loogikatehetel (`and`, `or`, `not`) on operandideks reeglina tõeväärtustüüpi väärtused (st *True* ja *False).* Tegelikult on olukord keerulisem – kõiki arve, mis ei võrdu nulliga, tõlgendatakse kui väärtust *True*. Kõige turvalisem (ja teiste keeltega ühilduvam) on siiski kasutada loogikatehetega koos tõeväärtustüüpi operande.

Tehete järjekord määratakse järgmiselt: kõigepealt täidetakse võrdlustehted vasakult paremale; nende tehete tulemuseks on alati loogikaväärtused. Nüüd saab loogikaväärtustele rakendada loogikatehteid. Ikka vasakult paremale.

Loogikaavaldistest tekkivaid tõeväärtusi on vaja kasutada näiteks valiku- (tingimus-) ja tsüklilausetes.

Näiteid avaldistest:

A on suurem 0-st ja väiksem/võrdne 10-st (ehk A on 1 ja 10 vahel):

```
A > 0 and A <= 10
```

Eelmisega samaväärne on Pythonis kirjaviis:

```
0 < A <= 10
```

Kuna aga sellist avaldise varianti enamus programmeerimiskeeli ei toeta, siis on targem õppida selgeks esimene variant):

A on 0-st väiksem või 10-st suurem (või ka A ei jää määratud vahemikku):

```
A < 0 or A > 10
```

On kokkulepitud, millised on erinevate loogikatehete tulemused. Et võimalikke väärtuseid on vaid kaks ja loogikatehteid ei ole ka palju, on selle jaoks koostatud nn tõeväärtustabelid.

and	True	False
True	True	False
False	False	False

or	True	False
True	True	True
False	True	False

not		
True	False	
False	True	

Näide loogikaavaldise väärtuse leidmisest

Avaldis on järgmine: **`A<0 or A>10`**

Läbimõtlemiseks võib anda muutuja(te)le konkreetsed väärtused ja neid kasutades loogikaavaldise tulemus leida. Andes A-le väärtuseks **5,** saame järgmise lause:

`5<0 or 5>10`

`5<0 - false; 5>10 – false`

`false or false` on vastavalt tõeväärtustabelile `false`

Andes A-le väärtuseks **12**, saame järgmise lause:

`12<0 or 12>10`

`12<0 - false; 12>10 – true`

`false or true` on vastavalt tõeväärtustabelile `true`

Olulist loogikaavaldistest

- loogikatehetel ja -muutujatel on vaid kaks võimalikku väärtust: `true` ja `false`
- võrdlustes kasutatavate operandide tüübid peavad kokku sobima
- ole ettevaatlik ujukomaarvude võrdlemisel eriti == - ga – võrrelda tuleks mingi täpsusega ehk mitu kohta peale koma peaksid võrdsed olema, et kaks arvu võrdseks tunnistada
- kasuta täiendavaid sulge, et muuta avaldis paremini arusaadavaks
- ära unusta tõeväärtustabeleid

Stringavaldised

Stringavaldise operandideks on stringid (ehk tekstitüüpi väärtused). Stringidele saab rakendada paari tehet – liitmist ja täisarvuga korrutamist. Lisaks on mitmeid funktsioone, kuid neid tutvustame edaspidi.

Näide:

```
eesnimi = "Maali"
perenimi = "Maasikas"
nimi = eesnimi + perenimi annab tulemuseks "MaaliMaasikas"
nimed = 3 * eesnimi annab tulemuseks "MaaliMaaliMaali"
```