CT10A0013 Ohjelmointi Pythonilla

L10: Data-analytiikka ohjelmoinnin näkökulmasta ja ohjelmointityylit

Uolevi Nikula

Päivän ohjelma

- Teoria
  - Data-analytiikan perusteet
  - Ohjelmointityyleistä
- Käytäntö
  - Sanakirja ja numpy-matriisi
  - Lajittelu
- Lopuksi

Teoria

Data-analytiikan perusteet
Ohjelmointityylit

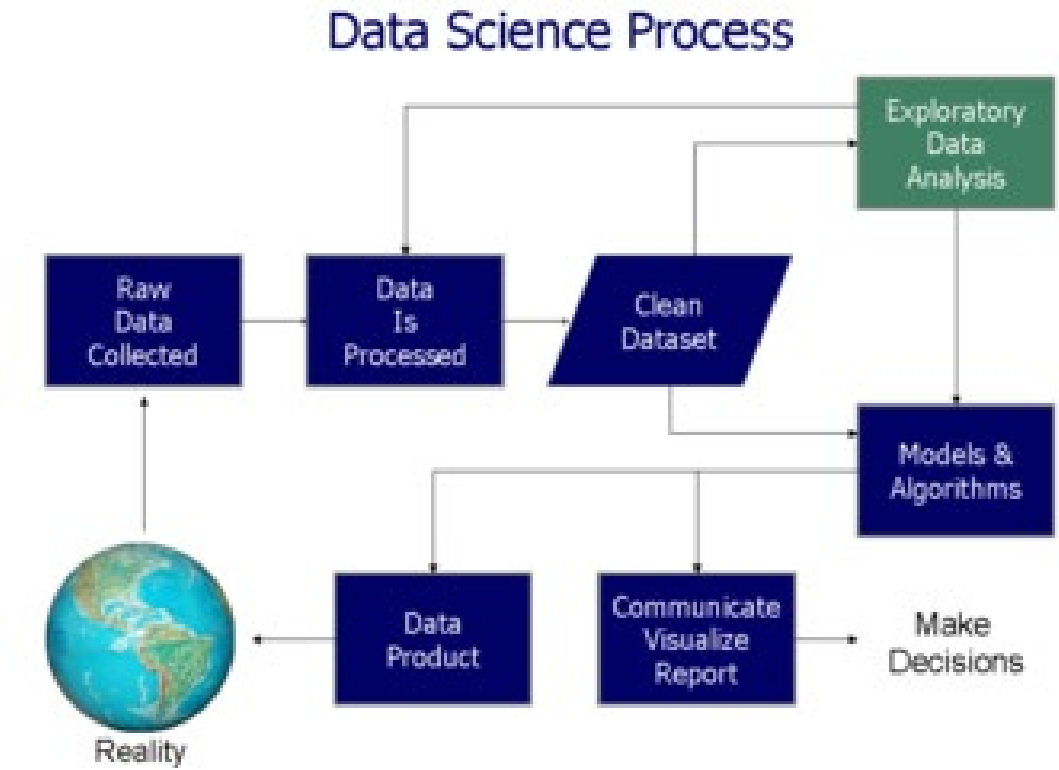
Mitä on data-analytiikka?

- Prof. Collan, strateginen laskentatoimi, business analytiikka
  - DA on osa tekoälyä
- Tämä kurssi eli Ohjelmoinnin perusteet
  - Data: tietokannat – MS-Access, MySQL, Oracle, MongoDB, …
  - DA: laskenta – Excel, R, **Python,** Pandas, …
  - Visualisointi: Excel, svgwrite, matplotlib, …
  - Tekoäly: päättelykone (jos … niin …) Lisp, Prolog, neuraaliverkot (NN, itseohjautuva päättely), …

Johdanto

- DA on prosessi datan tarkastamiseen, siivoamiseen, muuntamiseen ja mallintamiseen tavoitteena löytää hyödyllistä informaatiota, ehdottaa johtopäätöksiä ja tukea päätöksentekoa
- Käsitteitä on useita, ml. data science ja business analytics hieman erilaisilla painotuksilla
- Tämä johdanto keskittyy **ohjelmointinäkökulmaan**
  - Miten **raakadatasta** saadaan muokattua perusohjelmoinnin keinoilla **informaatiota**, jota voi käyttää johonkin **hyödylliseen**

Data Science Process (O'Neil 2013)



Prosessin vaiheet 1/2

1. Datan kerääminen
   - Esim. tietojärjestelmistä (Moodle, SISU, Traficom, …), antureilla säästä ja liikenteestä jne.
2. Datan prosessointi, käsittely
   - Muodostetaan raakadatasta analyysiin sopivaa aineistoa, tyypillisesti taulukoita, listoja, luokkia yms. helposti prosessoitavia yksiköitä
3. Datan siivous
   - Poistetaan puuttuvat ja virheelliset data-alkiot jne.
   - Tehdään tarkistuksia dataan: min/max arvot, summat, jne.
4. Exploratiivinen eli kokeileva analyysi
   - Lasketaan erilaisia (tilastollisia) tunnuslukuja ja arvioidaan tulosten kiinnostavuutta tavoitteena löytää kiinnostavia tietoja ja esitystapoja

Prosessin vaiheet 2/2

5. Mallinnus ja algoritmikehitys
   - Korrelaatiot, regressioanalyysit, …
6. Datatuoteen kehitys
   - Mikäli analyysiprosessi tuottaa kiinnostavia tuloksia, voidaan se automatisoida sopivalla ohjelmalla ja näin tehdä vastaavia analyysejä nopeasti, tehokkaasti ja luotettavasti – ***aina samalla tavalla***
7. Tulosten kommunikointi
   - Tyypillisesti tulosten visualisointi erilaisilla taulukoilla, graafeilla, diagrammeilla jne.

Harjoitustyö ja data-analytiikka

1. Data: dataa – sähkön hinta- ja kulutustietoja
2. Käsittely: dataa/tekstitiedostoja → luokat, oliot ja listat
3. Siivous: puuttuvat aineistot/data-alkiot, ongelmakentät – kiinnostava data
4. Exploratiivinen analyysi: erilaisia graafeja, eri datasettejä, kiinnostava / visuaalinen data
5. Mallinnus: ei tehdä
6. Datatuote: eri tiedostokoot, ohjelma, automatisointi – ei tehdä
7. Kommunikointi: visualisointi, graafit – Excelillä

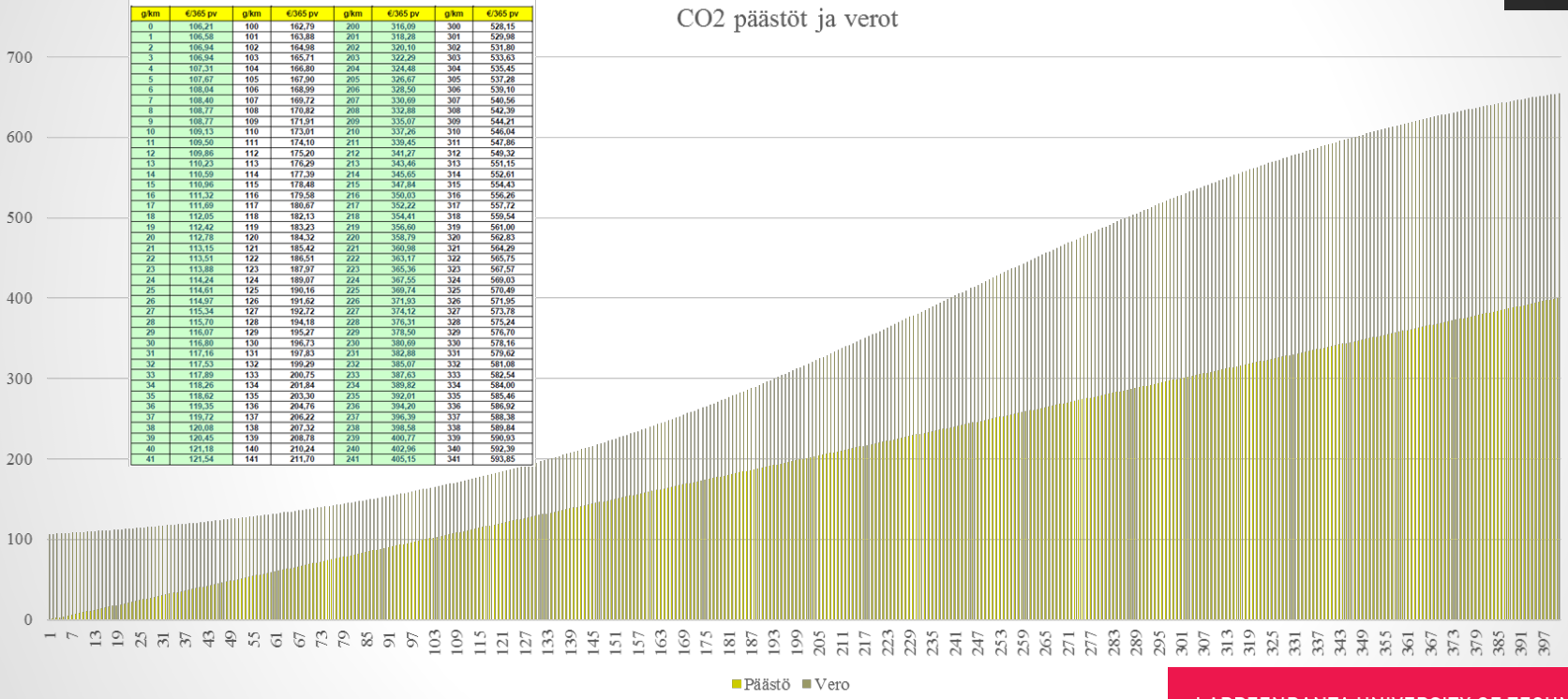
Tulosten visualisointitekniikoita

1. Aikasarjat, tietty ajanjakso, esim. viivadiagrammi
2. Rankkaus eli paremmuusjärjestys, nouseva tai laskeva, pylväsdiagrammi
3. Osa vs. kokonaisuus, pinta-alat, piirakkadiagrammi
4. Poikkeavuudet, suhteellinen vs. absoluuttinen, esim. pylväsdiagrammi
5. Taajuusjakauma, esim. esiintymismäärät eri vuosina eri kategorioissa, esim. histogrammi (pylväskaavio aloilla)
6. Korrelaatio, hajontakuvio (scatter plot)
7. Nominaalivertailu
8. Geograafinen tai paikkatietoon liittyvä, karttapohjalla

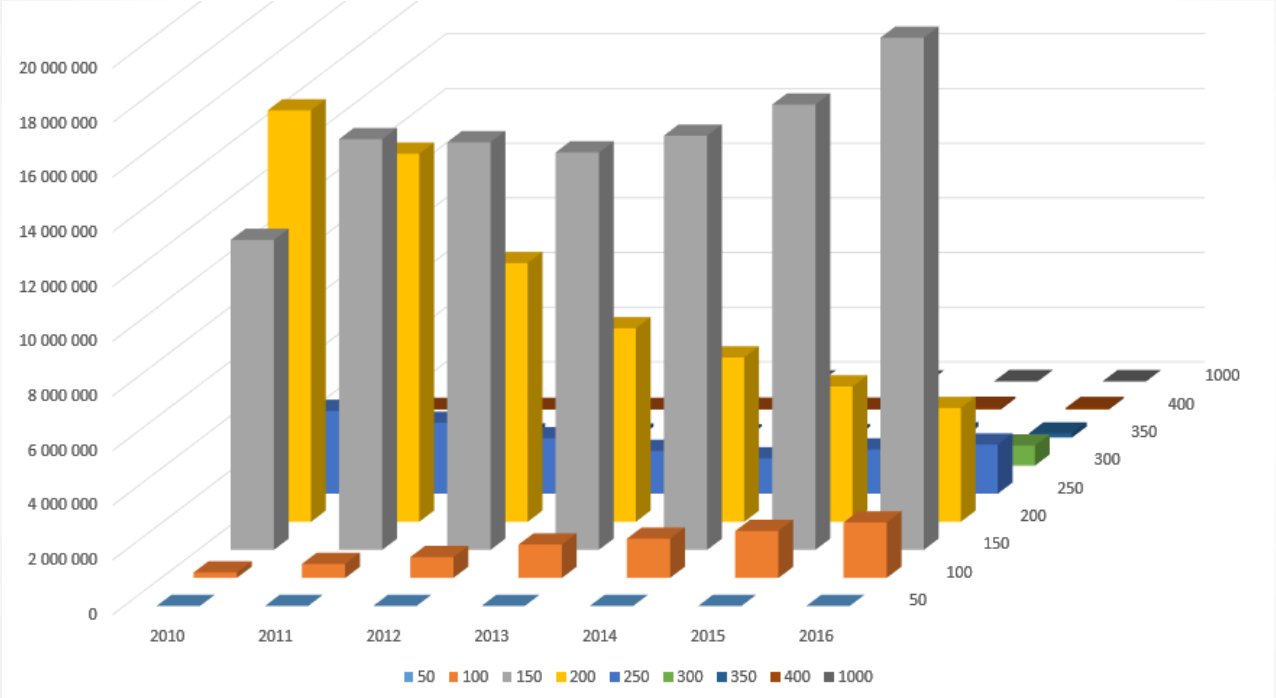
Datan visualisointi, CO2

Ajoneuvoveron perusvero hiilidioksidipäästön mukaan 1.1.2017-

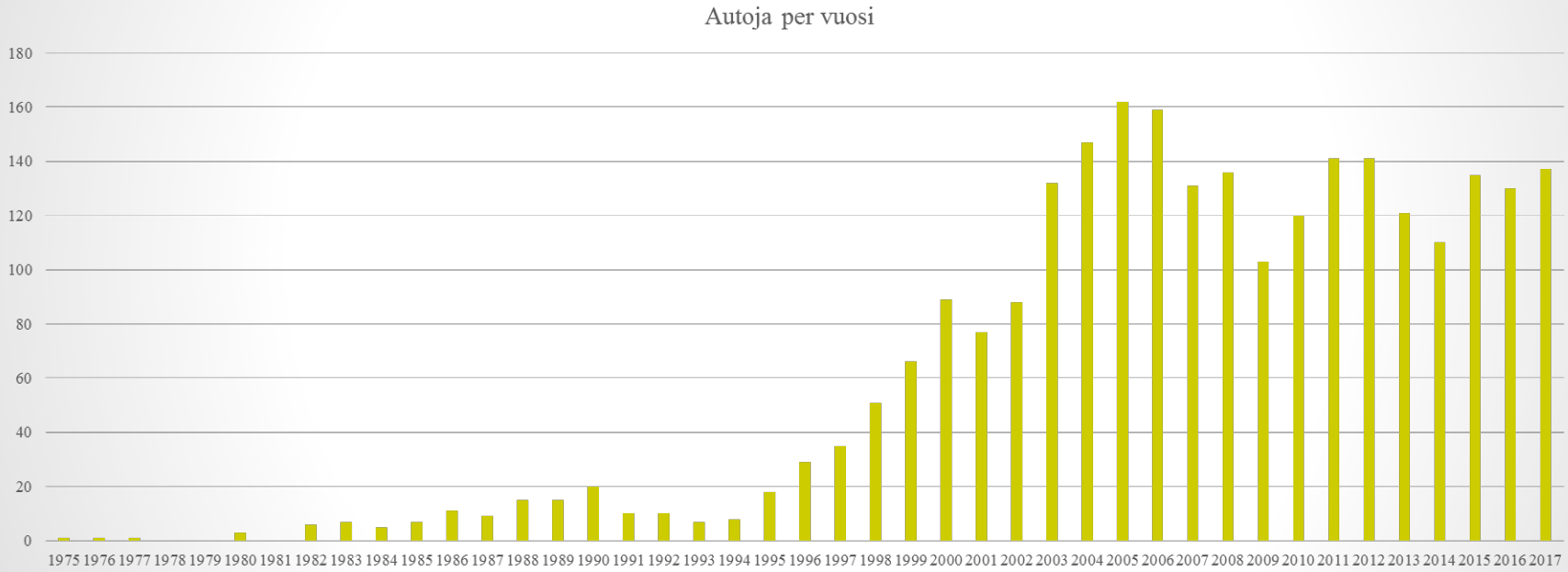
g/km	€/365 pv	g/km	€/365 pv	g/km	€/365 pv	g/km	€/365 pv
0	106,21	100	162,79	200	316,09	300	528,15
1	106,58	101	163,88	201	318,28	301	529,98
2	106,94	102	164,98	202	320,10	302	531,80
3	106,94	103	165,71	203	322,29	303	533,63
4	107,31	104	166,80	204	324,48	304	535,45
5	107,67	105	167,90	205	326,67	305	537,28
6	108,04	106	168,99	206	328,50	306	539,10
7	108,40	107	169,72	207	330,69	307	540,56
8	108,77	108	170,82	208	332,88	308	542,39
9	108,77	109	171,91	209	335,07	309	544,21
10	109,13	110	173,01	210	337,26	310	546,04
11	109,50	111	174,10	211	339,45	311	547,86
12	109,86	112	175,20	212	341,27	312	549,32
13	110,23	113	176,29	213	343,46	313	551,15
14	110,59	114	177,39	214	345,65	314	552,61
15	110,96	115	178,48	215	347,84	315	554,43
16	111,32	116	179,58	216	350,03	316	556,26
17	111,69	117	180,67	217	352,22	317	557,72
18	112,05	118	182,13	218	354,41	318	559,54
19	112,42	119	183,23	219	356,60	319	561,00
20	112,78	120	184,32	220	358,79	320	562,83
21	113,15	121	185,42	221	360,98	321	564,29
22	113,51	122	186,51	222	363,17	322	565,75
23	113,88	123	187,97	223	365,36	323	567,57
24	114,24	124	189,07	224	367,55	324	569,03
25	114,61	125	190,16	225	369,74	325	570,49
26	114,97	126	191,62	226	371,93	326	571,95
27	115,34	127	192,72	227	374,12	327	573,78
28	115,70	128	194,18	228	376,31	328	575,24
29	116,07	129	195,27	229	378,50	329	576,70
30	116,80	130	196,73	230	380,69	330	578,16
31	117,16	131	197,83	231	382,88	331	579,62
32	117,53	132	199,29	232	385,07	332	581,08
33	117,89	133	200,75	233	387,63	333	582,54
34	118,26	134	201,84	234	389,82	334	584,00
35	118,62	135	203,30	235	392,01	335	585,46
36	119,35	136	204,76	236	394,20	336	586,92
37	119,72	137	206,22	237	396,39	337	588,38
38	120,08	138	207,32	238	398,58	338	589,84
39	120,45	139	208,78	239	400,77	339	590,93
40	121,18	140	210,24	240	402,96	340	592,39
41	121,54	141	211,70	241	405,15	341	593,85



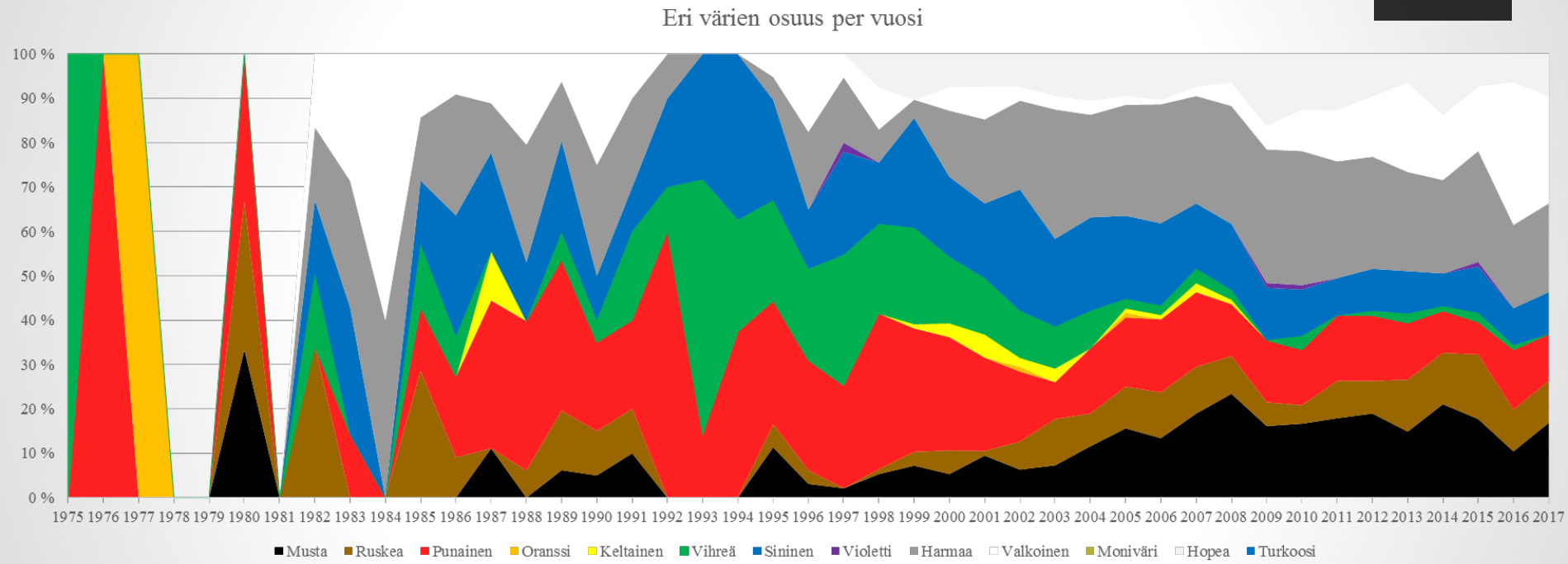
Ajoneuvoverokertymä



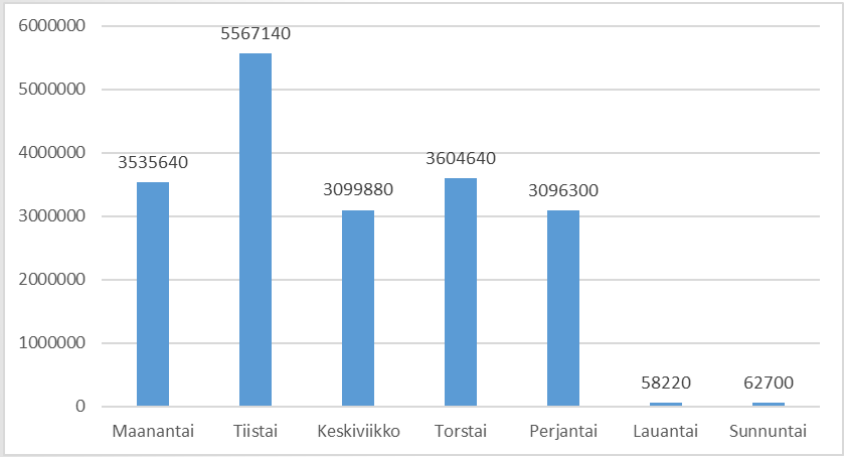
Osa Traficom’n 2018 ajoneuvotiedoista

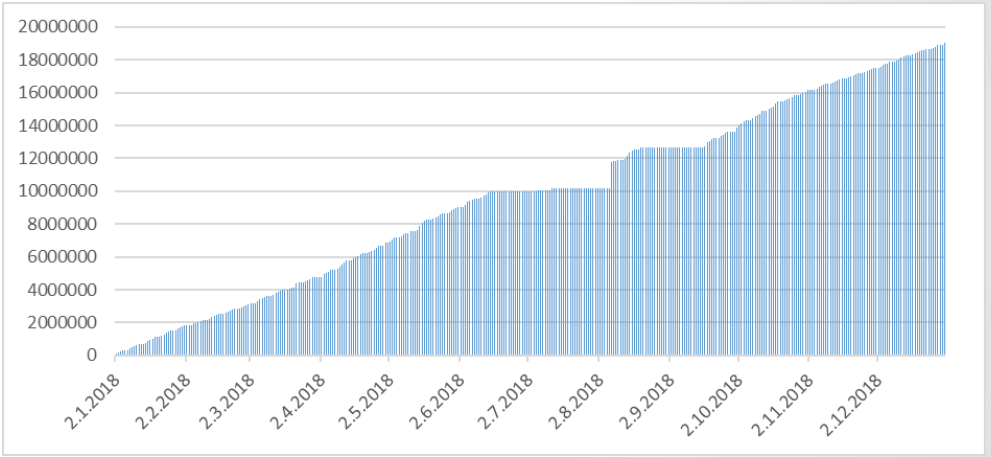


Osa Traficom’n 2018 ajoneuvotiedoista

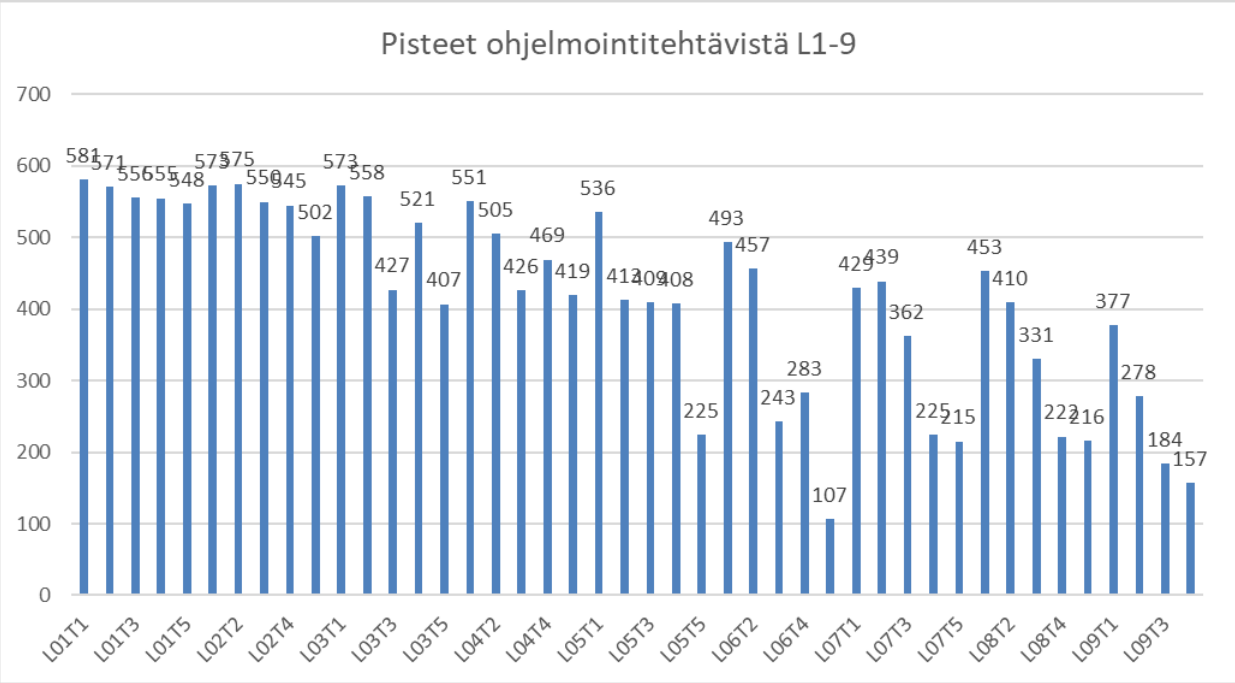


Jätekuormadataa





Viikkotehtävistä annetut pisteet L1-9



Datan visualisointi

- Animaatioita ilmastonmuutoksesta
  - Suomalaistutkija: "Ei näytä, että ilmaston lämpeneminen olisi vähenemässä”
  - https://twitter.com/anttilip/status/892318734244884480
    - https://yle.fi/uutiset/3-9754236
  - https://yle.fi/uutiset/3-10377870
- TED-talk, tilastotiedettä
  - Datan esittelyä, visualisointia, määrällisesti ja laadullisesti
  - https://www.ted.com/talks/hans_rosling_shows_the_best_stats_you_ve_ever_seen?language=fi

Ohjelmointityylit

Suunnittelulähtöinen vs. ketterä ohjelmakehitys

Ohjelmointityyleistä

- Ohjelmakehityksen perustehtävät ovat (eXtreme Programming, XP)
  - Määrittely
  - Suunnittelu
  - Toteutus
  - Testaus
- Näitä eri vaiheita voidaan tehdä useilla eri tavoilla
- Ohjelman (tai laajemmin ottaen ohjelmiston) kehityksen eri vaiheiden liittymistä toisiinsa kuvataan usein *ohjelmistokehitysprosessi*-termillä
  - Prosessi viittaa siihen, miten eri vaiheet liittyvät toisiin
  - Esimerkkejä prosessimalleista ovat vesiputousmalli ja spiraalimalli

Suunnitteluun perustuva ohjelmakehitys

- Perinteinen näkemys ohjelmakehityksestä on nk. vesiputousmalli
- Eri vaiheet tehdään peräkkäin ja periaatteessa ne etenevät vaiheesta toiseen eikä edelliseen vaiheeseen palata takaisin – vrt. vesiputous
- Periaatteessa ohjelma siis määritellään, suunnitellaan, toteutetaan, testataan ja otetaan käyttöön
- Käytännössä kaikkien asioiden etukäteen selvittäminen on usein vaikeaa eikä vesiputousmallin mukainen toiminta ole usein toimiva ratkaisumalli vaan aikaisempiin vaiheisiin joudutaan usein palaamaan
  - Erityisesti aloittelijan on vaikea suunnitella kaikki asiat valmiiksi etukäteen ja sitten vain toteuttaa ohjelma kerralla valmiiksi

Ketterä ohjelmakehitys

- Ketterä ohjelmistokehitys (agile software development) on vahvistanut asemiaan 2000-luvulla
- Lähestymistapa painottaa ohjelman kehitystä vähän kerrallaan vaiheittain
- Käytännössä tehdään yksi toiminto "valmiiksi" eli määritellään, suunnitellaan, toteutetaan ja testataan yksi toiminto kerrallaan
- Yhden toiminnon valmistuttua siirrytään seuraavaan
- Ohjelmistokehitysmalleihin liittyvät termit *iteratiivinen* ja *evolutionäärinen* kuvaavat hyvin tämän lähestymistavan luonnetta

Ohjelmointityylin valinnasta 1

- Suunnitteluun perustuva malli (top-down)
  - Top-down lähestymistapa johtaa yleensä kokonaisvaltaisesti suunniteltuun ratkaisuun
  - Vähentää usein koodin määrää, koska usein toistuvat toiminnot on helpompi tunnistaa ja toteuttaa yhteisinä komponentteina, esim. aliohjelmina
  - Edellyttää asioiden ymmärtämistä ja osaamista, jotta voi valita oikeat ratkaisut etukäteen
    - Osaaminen ja ymmärrys kehittyvät kokemuksen ja ajan myötä

Ohjelmointityylin valinnasta 2

- Ketterä ohjelmakehitys (usein bottom-up)
  - Bottom-up lähestymistapa mahdollistaa toimivan ohjelman tekemisen nopeasti
  - Lähtökohta on toteuttaa ensin se toiminnallisuus mitä osaa, ja sitten siirtyä yhteen uuteen ongelmaan kerrallaan
  - Sopii erityisesti pientien ohjelmien tekemiseen, koska kokonaisvaltaisen näkemyksen merkitys ei kasva suureksi
  - Johtaa helposti koko ohjelman ajoittaiseen uudelleen organisointiin (refactoring), sillä alussa tuntemattomat tai huomiotta jätetyt vaatimukset saattavat tehdä uuden toiminnallisuuden lisäämisestä mahdotonta
    - Esim. aiemmin yhdellä muuttujalla toteutettu ratkaisu pitää vaihtaa dynaamiseen ratkaisuun (esim. listaan), jotta pystytään käsittelemään useita samanlaisia tietoja

Ohjelmointityylin valinnasta 3

- Tällä kurssilla vallitseva ohjelmointityyli on ketterä ja kokeileva tyyli
- Sekä luentodemot että harjoitukset sopivat tyypillisesti tällaiseen kokeilevaan tyyliin
- Koko kurssi on periaatteessa suunniteltu top-down menetelmällä kurssin yleisistä vaatimuksista lähtien
- Kuten uudelleenkäytön yhteydessä oli puhetta, suurimpia ongelmia uudelleenkäyttöön liittyen on saatavilla olevien kirjastojen huono tuntemus
  - Suunnitteluun perustuva ohjelmakehitys ja uudelleenkäyttö sopivat hyvin yhteen

Koodiesimerkkejä

Sanakirja
numpy-matriisi
Lajittelu

Lista vs. sanakirja

- Lista-tietorakenne
  - Ei ota kantaa sisältöön millään tavalla, data yksinkertaisesti *lisätään* listaan
  - Tiedon *haku* perustuu **järjestykseen**, esim. haetaan 3. elementti indeksillä 2, tai käydään kaikki data-alkiot läpi ja tunnistetaan haluttu alkio, esim. if (alkio.nimi == “Ville”) …
  - Tiedon lisääminen nopeaa, mutta haku hidasta ja riippuu datan määrästä
- Sanakirja-tietorakenne
  - Sanakirjaan alkiot *lisätään* **avain-data –pareina**
  - Tämän seurauksena data-alkio voidaan *hakea* **avaimen perusteella** suoraan sanakirjasta. Alkioilla ei ole “järjestystä”, sillä ne sijoitetaan sanakirjaan sisäiseen tietorakenteeseen matemaattisen kaavan avulla eli data haetaan aina avaimesta lasketun tiedon perusteella
  - Tiedon lisääminen ja haku vie aina laskentaan perustuvan vakioajan

- Huom. Tuple = (); Lista = []; Sanakirja = {}

Sanakirja: avain ja data -pareja

```
Sahkoposti = {"167-671" : "bigtime@beagle.biz",
              "176-761" : "burger@beagle.biz" }
print("167-671 sähköposti on", Sahkoposti["167-671"])

# Sanakirjan määrittely
Sanat={"avain1" : "arvo1", "avain2" : "arvo2", 22:12}
# Sanakirjan käyttö
print(Sanat["avain1"])
print(Sanat[22])
```

numpy-matriisi: ks. Opas matriisiA[i][j]

```
import numpy

MatriisiA = numpy.array( # Matriisin luominen, alustus halutuilla arvoilla
    [[1,2,4],
     [5,6,7],
     [8,9,10]])

# Laskentaa matriiseilla numpyn avulla
MatriisiB = MatriisiA + MatriisiA

# Matriisin tulostaminen numpyn avulla
print(MatriisiB)
# Matriisin tuhoaminen, rivin/sarakkeen/alkion poistaminen mahdollista
MatriisiA = numpy.delete(MatriisiA, numpy.s_[:], None)
```

Lajittelu: tuple, sanakirja, olio-lista

```
Tuple = (1,10,2,7,12) # yksiulotteinen tuple, vrt. lista
print(sorted(Tuple))

Opiskelijat = (       # moniulotteinen tuple, vrt. matriisi
        ('Matikainen', 'A.', 15),
        ('Asikainen', 'K.', 12),
        ('Viippola', 'S.', 13)
        )

Lajiteltu=(sorted(Opiskelijat, key=lambda Opiskelijat:Opiskelijat[0]))
print(Lajiteltu)

# Huom. Yo. opiskelijat-tuple sisältää tupleja, ts. alkioissa on useita alkioita
# Sanakirjassa on aina alkiopari, avain ja data
# Olio-listan alkioissa, olioissa, voi olla monta alkiota eli jäsenmuuttujaa
```

Lopuksi

Osaamistavoitteet

Osaamistavoitteet

- Data-analytiikan perusteet Pythonilla
  - Teksti-pohjaisten data-tiedostojen lukeminen ja tallentaminen sopivaan tietorakenteeseen analyysia varten, tyypillisesti luokka, olio, lista, sanakirja, matriisi
  - Tiedon muokkaus siten, että tiettyjä asioita voidaan korostaa, esim. lajittelemalla tai valitsemalla tiettyjä alkioita tms.
  - Tiedon tallennus sellaisessa muodossa, että sen jatkojalostaminen on helppoa toisilla työkaluilla, esim. kuvaajien piirtäminen Excelillä
- Suunnittelulähtöinen vs. ketterä ohjelmistokehitys
- Ohjelmointi
  - Sanakirja
  - numpy-matriisi
  - Lajittelu – sanakirja, tuple, oliolista

Täydennyksiä oppaan lukuun 10

Tyyliohjeita pienille Python-ohjelmille
ASPAssa ei enää uusia tarkistuksia
Oppaan esimerkit ja käsitellyt asiat

Pienen Python-ohjelman tyyliohjeet

- Tietyt tietotyypit pitää määritellä ennen käyttöä, esim. sanakirja, lista ja numpy-matriisi. Nämä tulee määritellä ja tyhjentää/vapauttaa samassa aliohjelmassa. Näin muistin vapautus on helppo varmistaa
- Edellä oli L08T5 tarkastuksen palautteita
  - Nämä kannattaa katsoa ja välttää niiden avulla samat virheet/ongelmat

Käsitellyt asiat oppaan luvussa 10

- Sanakirja: Esimerkki 10.1,
- Lajittelu: Esimerkki 10.2, 10.3
- numpy-matriisi: Esimerkki 10.4, 10.5
- Data-analytiikka

- **Kokoava esimerkki 10.5: numpy-matriisin käyttö datetime-kirjaston, kiintoarvojen, luokan ja poikkeustenkäsittelyn kanssa**