

Oplossingen Analyse Telefoongesprekken ¶

Deze notebook bevat de oplossingen van de vragen in de notebook "Extra oefening samengestelde datatypes: analyse telefoongesprekken". Lees eerst die notebook en probeer zelf de vragen op te lossen voordat je hier verder leest.

Het bestand "call.txt" bevat een lijst van paren van namen. Dit zijn beller en opgebelde in een telefoongesprek. Voer volgende lijnen code uit om het bestand in te lezen:

```
In [1]: def leesCallsIn():
        f=open("calls.txt", "r")
        pairs=[]
        for line in f:
            if len(line)==0:
                continue
            if line[-1]=='\n':
                line=line[:-1]

            pair=line.split()
            tuple=(pair[0],pair[1])
            pairs.append(tuple)
        f.close()
        return pairs

pairs=leesCallsIn()
```

Dit was het startpunt van de opgaves. Verwijder de commentaar en voer volgende lijn uit om je te beperken tot de eerste 8 gesprekken; dit was zo voorzien in de opgaves om debugging te vereenvoudigen.

```
In [2]: #pairs=pairs[:8] # Voer deze lijn uit om het aantal gesprekken te beperken tot de eerste 8.
```

Hieronder volgen de correcte antwoorden. Mogelijk bereik je hetzelfde antwoord op een andere manier. In het geval je twijfelt aan de correctheid van jouw stukje code, of je de fout in jouw stukje code niet begrijpt, contacteer dan de docent of een van de assistenten van het vak.

1. Hoeveel personen zijn er in totaal?

```
In [3]: personen=set()    # we gebruiken een set om duplicaten automatisch te vermijden
        for tuple in pairs:
            beller=tuple[0]
            opgebelde=tuple[1]
            personen.add(beller)
            personen.add(opgebelde)

        print(len(personen))
        #print("De personen zijn:",personen)
```

67

Je kan ook rechtstreeks tijdens de *for*-loop het tuple "uitpakken":

```
In [4]: personen=set()
        for (beller,opgebelde) in pairs:
            personen.add(beller)
            personen.add(opgebelde)

        print(len(personen))
        #print("De personen zijn:",personen)
```

67

1. Wat is het totale aantal gesprekken en wat is het aantal *unieke* gesprekken. Bij unieke gesprekken gaan we twee gesprekken met dezelfde beller en opgebelde slechts 1 maal tellen. Een gesprek van *a* naar *b* is echter wel een ander gesprek dan een gesprek van *b* naar *a*.

```
In [5]: print("Aantal gesprekken:",len(pairs))
        uniekeGesprekken=set(pairs)
        print("Aantal unieke gesprekken:",len(uniekeGesprekken))
```

Aantal gesprekken: 10008
Aantal unieke gesprekken: 3983

1. Welke personen waren bij het grootste aantal gesprekken betrokken (als beller of als opgebelde) en bij hoeveel gesprekken is dat ?

```
In [6]: aantalGesprekkenVan={}
for beller,opgebelde in pairs:
    if not beller in aantalGesprekkenVan: # key bestaat nog niet
        aantalGesprekkenVan[beller]=0 # maak hem aan en associeer met
    0
    if not opgebelde in aantalGesprekkenVan: # key bestaat nog niet
        aantalGesprekkenVan[opgebelde]=0 # maak hem aan en associeer m
    et 0
    aantalGesprekkenVan[beller]+=1 # verhoog met 1 voor beller ...
    aantalGesprekkenVan[opgebelde]+=1 # ... idem voor opgebelde

# We hebben van iedereen nu het aantal keer dat die in een gesprek betrokken w
as
# Nu moeten we nog het maximum zoeken:
maximumTotNu=0
personenMetMaximum=set()
for persoon in aantalGesprekkenVan:
    if aantalGesprekkenVan[persoon]==maximumTotNu:
        personenMetMaximum.add(persoon)
    elif aantalGesprekkenVan[persoon]>maximumTotNu:
        maximumTotNu=aantalGesprekkenVan[persoon]
        personenMetMaximum = {persoon}

print("Maximum aantal gesprekken is",maximumTotNu,"en werd bereikt door",perso
nenMetMaximum)
```

Maximum aantal gesprekken is 333 en werd bereikt door {'Stephen'}

Merk op dat we een dictionary vaak gebruiken om frequenties te tellen. Daarbij is de volgende constructie heel populair:

1. maak een lege dictionary `d`
2. een nieuw element `e` moet geteld worden
3. is `e` al een key in `d` ?
 - Indien neen: voeg de key toe aan `d` , met geassocieerde waarde `0` : `d[e]=0`
4. verhoog de waarde `d[e]` met 1: `d[e]=d[e]+1`

De test op lijn 3 is noodzakelijk omdat anders voor een nieuw element `e` , regel 4 niet kan uitgevoerd worden omdat `d[e]+1` moet berekend worden en de key `e` niet gekend is. We krijgen dan een `keyerror`:

```
In [7]: d={} # lege dictionary
d['a']=d['a']+1
```

```
-----
KeyError                                Traceback (most recent call last)
<ipython-input-7-b73b205abf19> in <module>
      1 d={} # lege dictionary
----> 2 d['a']=d['a']+1

KeyError: 'a'
```

Daarom maken we gebruik van volgend stukje code:

```
In [8]: d={} # lege dictionary
        if not 'a' in d:
            d['a']=0
        d['a']=d['a']+1
```

Omdat deze constructie zodanig veel voorkomt werd aan dictionaries de method `get(key,default)` toegevoegd die de waarde geassocieerd met `key` teruggeeft, behalve als die key niet bestaat; dan geeft die `default` terug. Bovenstaand stukje code kan dan herschreven worden als:

```
In [9]: d={} # lege dictionary
        d['a']=d.get('a',0)+1
```

De eerste 8 lijnen van de code voor de oplossing van deze vraag kan dus herschreven worden als volgt:

```
In [10]: aantalGesprekkenVan={}
        for beller,opgebelde in pairs:
            #if not beller in aantalGesprekkenVan: # key bestaat nog niet
            #    aantalGesprekkenVan[beller]=0      # maak hem aan en associeer met
            0
            #if not opgebelde in aantalGesprekkenVan: # key bestaat nog niet
            #    aantalGesprekkenVan[opgebelde]=0      # maak hem aan en associeer
            met 0
            #aantalGesprekkenVan[beller]+=1      # verhoog met 1 voor beller ...
            #aantalGesprekkenVan[opgebelde]+=1 # ... idem voor opgebelde
            aantalGesprekkenVan[beller]=aantalGesprekkenVan.get(beller,0)+1
            aantalGesprekkenVan[opgebelde]=aantalGesprekkenVan.get(opgebelde,0)+1
```

1. Welke persoon werden het minste opgebeld ? Hoeveel keer was dat ?

Zorg ervoor dat oplossing voor vraag 1 werd uitgevoerd, want we hebben een set met alle personen nodig! Die werd in cel met de oplossing van vraag 1 berekend en in de variabele personen bewaard.

```
In [11]: # eerst bekijken we hoeveel maal iedereen opgebeld werd
# omdat dit 0 kan zijn (!!!) maken we eerst al voor elke persoon een key aan i
n de dictionary
# 0 als aantal maal opgebeld.
aantalOpgebeld={}
for persoon in personen:
    aantalOpgebeld[persoon]=0

# nu gaan we over alle gesprekken en verhogen de teller van de opgebelde
for beller,opgebelde in pairs:
    aantalOpgebeld[opgebelde]+=1 # we hebben de constructie met get hier niet
nodig want de key bestaat zeker!

# we hebben nu per persoon hoe vaak die opgebeld werd. Nu moeten we berekenen
wie het minste werd opgebeld
minimumTotNu=len(pairs) # zorg ervoor dat dit groot genoeg is!
personenMetMinimum=set()
for persoon in aantalOpgebeld:
    if aantalOpgebeld[persoon]==minimumTotNu:
        personenMetMinimum.add(persoon)
    elif aantalOpgebeld[persoon]<minimumTotNu:
        personenMetMinimum={persoon}
        minimumTotNu=aantalOpgebeld[persoon]

print("Minst aantal keer gebeld is",minimumTotNu,"voor personen",personenMetMi
nimum)
```

Minst aantal keer gebeld is 114 voor personen {'Jens'}

1. Hoeveel maal wordt iemand gemiddeld opgebeld ?

Zorg dat de code voor vraag 4 werd uitgevoerd, want we hebben de dictionary `aantalOpgebeld` nodig.

```
In [12]: som=0
for persoon in aantalOpgebeld:
    som+=aantalOpgebeld[persoon]
print("Gemiddeld aantal maal opgebeld is:",som,"/",len(personen),("(",som/len(p
ersonen),")"))
```

Gemiddeld aantal maal opgebeld is: 10008 / 67 (149.37313432835822)

1. Welke personen hadden contact met het grootste aantal andere mensen (als *a*, *b* opbelt, heeft *a* contact gehad met *b* en omgekeerd. Het maakt bij deze vraag niet uit hoe vaak *a*, *b* opbelde). Hoeveel contactpersonen hebben deze mensen?

```
In [13]: contactpersonenVan={}
for beller,opgebelde in pairs:
    if not beller in contactpersonenVan: # als de key beller nog net bestaat
        contactpersonenVan[beller]=set() # voeg hem dan toe en associeer met
een lege verzameling
    if not opgebelde in contactpersonenVan: # als de key beller nog net bestaat
        contactpersonenVan[opgebelde]=set() # voeg hem dan toe en associeer
met een lege verzameling
    contactpersonenVan[beller].add(opgebelde) # opgebelde is contactpersoon van beller
    contactpersonenVan[opgebelde].add(beller) # en omgekeerd

# Nu hebben we per persoon de contactpersonen; nu moeten we enkel nog diegenen
met het grootste aantal vinden
maximumTotNu=0
personenMetMaximum=set()
for persoon in contactpersonenVan: # itereer over alle keys
    if len(contactpersonenVan[persoon]) == maximumTotNu:
        personenMetMaximum.add(persoon)
    elif len(contactpersonenVan[persoon]) > maximumTotNu:
        maximumTotNu=len(contactpersonenVan[persoon])
        personenMetMaximum={persoon}

print("Maximum aantal contactpersonen is",maximumTotNu,"en dit wordt bereikt door",personenMetMaximum)
```

Maximum aantal contactpersonen is 66 en dit wordt bereikt door {'Daniel', 'Zlatko', 'Felix', 'Mano', 'Arthur', 'Noah', 'Isabelle', 'Jeff', 'Sander', 'Lars', 'Aymane', 'Edmond', 'Alexander', 'Stephen', 'Omar', 'Recep', 'Tom', 'Mohamed', 'Saartje', 'Jord', 'Jasper', 'Lukas', 'Manjock', 'Bas', 'Arno', 'Giorgi', 'Joshua', 'Erhan', 'Niels', 'Khalil', 'Tycho', 'Rafik', 'Ferit', 'Eliza'}

1. Welke paren van mensen hadden het meeste contact met elkaar? Hoe vaak hadden deze personen contact?

Bij deze vraag moeten we paren tellen. Merk op dat de paren (a,b) en (b,a) als hetzelfde paar worden beschouwd. Het is dan ook verleidelijk om een dictionary te maken met de set {a,b} als key. Dit is echter niet mogelijk in Python, omdat sets *mutable* zijn (we zien later wat dit betekent). Daarom zullen we een van de tupels (a,b) en (b,a) kiezen en daarmee werken om het paar voor te stellen. Om dubbele tellingen te voorkomen zullen we het paar voorstellen door het tuple waarin de namen in alfabetische volgorde staan. We kunnen twee strings s1 en s2 vergelijken met behulp van s1<s2 ; dit is True als s1 voor s2 komt in alfabetische volgorde.

```

In [14]: aantalContactenVan={} # Lege dictionary
for beller,opgebelde in pairs:
    if beller<opgebelde: # zorg ervoor dat paar de namen in alfabetische volg
orde bevat
        paar=(beller,opgebelde)
    else:
        paar=(opgebelde,beller)
    aantalContactenVan[paar]=aantalContactenVan.get(paar,0)+1 # verhoog de t
eller van paar

# We hebben nu alle paren en het aantal keer dat ze elkaar belden. Nu moeten w
e het maximum berekenen
maximumTotNu=0
parenMetMaximum=set()
for paar in aantalContactenVan:
    if aantalContactenVan[paar]==maximumTotNu:
        parenMetMaximum.add(paar)
    elif aantalContactenVan[paar]>maximumTotNu:
        parenMetMaximum={paar}
        maximumTotNu=aantalContactenVan[paar]
print("Maximaal aantal contacten:",maximumTotNu,"voor paren",parenMetMaximum)

```

Maximaal aantal contacten: 15 voor paren {('Arbesa', 'Thibault')}

1. Bij welke mensen is het verschil tussen gevoerde en ontvangen gesprekken het grootst? Hoe groot is dit verschil ? Hoe vaak werden deze personen zelf gebeld, respectievelijk belden ze zelf?

```

In [15]: # We maken tellers voor opgebeld en zelfgebeld voor alle personen
# we initialiseren die op 0
opgebeld={}
zelfgebeld={}
for persoon in personen:
    opgebeld[persoon]=0
    zelfgebeld[persoon]=0

# Nu gaan we over alle gesprekken en tellen:
for beller,opgebelde in pairs:
    opgebeld[opgebelde]=opgebeld.get(opgebelde,0)+1
    zelfgebeld[beller]=zelfgebeld.get(beller,0)+1

# en nu moeten we het grootste verschil zoeken
grootsteVerschilTotNu=0
personenMetGrootsteVerschil=set()
for persoon in personen:
    verschil=abs(opgebeld[persoon]-zelfgebeld[persoon])
    if verschil==grootsteVerschilTotNu:
        personenMetGrootsteVerschil.add(persoon)
    elif verschil>grootsteVerschilTotNu:
        grootsteVerschilTotNu=verschil
        personenMetGrootsteVerschil={persoon}

# rapporteren:
print("Het grootste verschil is",grootsteVerschilTotNu,"en werd bereikt door:"
)
for persoon in personenMetGrootsteVerschil:
    print(persoon,"(zelf gebeld:",zelfgebeld[persoon],", opgebeld:",opgebeld[p
ersoon],")")

```

Het grootste verschil is 52 en werd bereikt door:
 Mohamed (zelf gebeld: 185 , opgebeld: 133)