Forelesning IN1900 – 3 okt 2023

Ole Christian Lingjærde
Institutt for Informatikk, Universitetet i Oslo

Uke: 2 Oktober – 8 Oktober, 2023

Denne ukens agenda

Dictionaries

Oppgave A.14 og 5.14 i Langtangens bok

Tekststrenger

Oppvarming til midtveiseksamen

Introduksjon

Dictionary: en samling *regler* som knytter *nøkler* til *verdier*. Eksempel:

$d = \{0{:}6,\ 1{:}3,\ 2{:}7\}$

- Nøkler: 0, 1, 2
- Verdier: 6, 3, 7
- Regler: $0 \rightarrow 6$, $1 \rightarrow 3$, $2 \rightarrow 7$

Analogi til funksjoner

Anta at du ønsker å implementere disse reglene:

```
'Norway' --> 'Oslo'
'UK' --> 'London'
'France' --> 'Paris'
```

A) Implementasjon med funksjon:

```
def f(x):
    if x == 'Norway':
        return 'Oslo'
    elif x == 'UK':
        return 'London'
    elif x == 'France':
        return 'Paris'
```

B) Implementasjon med dictionary:

```
d = {'Norway':'Oslo', 'UK':'London', 'France':'Paris'}
```

En dictionary kan utvides med nye regler

Anta at vi i forrige eksempel ønsker *en ny regel*:

```
'Norway' --> 'Oslo'
'UK' --> 'London'
'France' --> 'Paris'
'Nepal'  --> 'Kathmandu'   # NY REGEL
```

Implementasjon med funksjon:
Må skrive en helt ny funksjon.

Implementasjon med dictionary:
Kan enkelt utvide en dictionary med nye regler:

```
d['Nepal'] = 'Kathmandu'
```

Analogi til lister

Nøkler i en dictionary = indekser i en liste

Merk: indeksene i lister er alltid 0, 1, ..., N-1, mens nøklene i en dictionary kan være nesten hva som helst.

Eksempler:

```
# Parene (-2,6), (6,3), (3,7)
d = {-2:6, 6:3, 3:7}

# Parene ('Hamar','Norway'), ('Uppsala','Sweden')
d = {'Hamar':'Norway', 'Uppsala':'Sweden'}

# Parene ('pi', 3.14), ('e', 2.718), ('g', 9.81)
d = {'pi':3.14, 'e':2.718, 'g':9.81}

# Parene ((0,0), 'lowerleft') and ((1,1), 'upperright')
d = {(0,0):'lowerleft', (1,1):'upperright'}
```

Nøklene i en dictionary må være 'immutable'

Nøklene i en dictionary må være immutable, dvs objekter som ikke kan endres.

Immutable:

- int-verdier og float-verdier
- string-verdier
- tupler

Ikke immutable:

- lister (kan endre element med a[i]=y)
- objekter av egendefinerte klasser

Viktig å huske

To regler kan ikke ha samme nøkkel:

Vi kan ikke skrive `d = {'Oslo': 3, 'Oslo': 4}`.

Vi kan skrive `d = {'Oslo': 3}` og så `d['Oslo'] = 5`, men da overskrives den første av de to.

To regler kan ha samme verdi:

Vi kan godt skrive `d = {'Oslo': 3, 'Bergen': 3}`.

Reglene har ingen ordning:

En dictionary lagrer ikke reglene i en bestemt rekkefølge. Tenk på en dictionary som en 'sekk med regler'.

Quiz

Er dette en dictionary, og hva er isåfall nøkkel og verdi?

```
d = {3;8}
```

Quiz

Er dette en dictionary, og hva er isåfall nøkkel og verdi?

```
d = {3;8}
```

Det er ikke en dictionary (skal stå kolon, ikke semikolon)

Quiz

Er dette en dictionary, og hva er isåfall nøkkel og verdi?

```
d = [3:8]
```

Quiz

Er dette en dictionary, og hva er isåfall nøkkel og verdi?

```
d = [3:8]
```

Det er ikke en dictionary (feil type parenteser)

Quiz

Er dette en dictionary, og hva er isåfall nøkkel og verdi?

```
d = {3:8}
```

Quiz

Er dette en dictionary, og hva er isåfall nøkkel og verdi?

```
d = {3:8}
```

Det er en dictionary, med nøkkel 3 og verdi 8.

Quiz

Er dette en dictionary, og hva er isåfall nøkkel og verdi?

```
d = {8:3, 5:3}
```

Lovlig dictionary med nøkler 8 og 5, og tilhørende verdier 3 og 3.

Quiz

Er dette en dictionary, og hva er isåfall nøkkel og verdi?

```
d = {3:8, 3:5}
```

Quiz

Er dette en dictionary, og hva er isåfall nøkkel og verdi?

```
d = {3:8, 3:5}
```

Lovlig dictionary, men den første regelen overskrives av den siste, siden nøklene er like. Dermed: nøkkel 3 og verdi 5.

Quiz

Er dette en dictionary, og hva er isåfall nøkkel og verdi?

```
d = {('A','A'): 'homozygot', ('A','B'): 'heterozygot'}
```

Quiz

Er dette en dictionary, og hva er isåfall nøkkel og verdi?

```
d = {('A','A'): 'homozygot', ('A','B'): 'heterozygot'}
```

Lovlig dictionary, med nøkler

`('A','A')` og `('A','B')`

og tilhørende verdier

`'homozygot'` og `'heterozygot'`

Quiz

Er dette en dictionary, og hva er isåfall nøkkel og verdi?

```
d = {['A','A']: 'homozygot', ['A','B']: 'heterozygot'}
```

Quiz

Er dette en dictionary, og hva er isåfall nøkkel og verdi?

```
d = {['A','A']: 'homozygot', ['A','B']: 'heterozygot'}
```

Feilmelding siden nøklene er lister:

```
Traceback (most recent call last):

  Input In [8] in <cell line: 1>
    d = {['A','A']: 'homozygot', ['A','B']:
'heterozygot'}

TypeError: unhashable type: 'list'
```

Å lage en dictionary

Initialisere en dictionary:

```
# Lag en tom dictionary:
d = {}
# Lag en dictionary med to par:
d = {'Oslo': 13, 'London': 15.4}
# Lag en dictionary med to par (alternativ metode):
d = dict(Oslo=13, London=15.4)
```

Utvide en dictionary:

```
# Legg et nytt par til d:
d['Madrid'] = 26.0
# Legg en hel dictionary d2 til d:
d.update(d2)
```

Å fjerne regler fra en dictionary

Fjerne et par (a,b) fra en dictionary:

```
d = {'pi':3.14, 'e':2.718, 'g':9.81}
del d['e']              # Fjern paret ('e', 2.718)
del d['pi']             # Fjern paret ('pi', 3.14)
```

NB: forsøk på å fjerne et par (a, ...) fra en dictionary gir feilmelding dersom ikke nøkkelen a finnes.

Teste om en nøkkel finnes i en dictionary

key in d:

Tester om key er en nøkkel i d

d[key]:

Henter ut verdien i d lagret under nøkkelen key.
Gir feilmelding hvis key ikke er en nøkkel i d.

d.get(key):

Henter ut verdien i d lagret under nøkkelen key.
Returnerer None hvis key ikke er en nøkkel i d.

Eksempel

```
d = {'Berkeley': 'US', 'Cambridge': 'UK'}
key = 'Berkeley'
if key in d:
    print(f'Regelen {key} --> {d[key]} ligger i dictionary')
else:
    print(f'Ingen regel med nøkkel {key} i dictionary')
```

Eksempel

```
d = {'Berkeley': 'US', 'Cambridge': 'UK'}
key = 'Berkeley'
if key in d:
    print(f'Regelen {key} --> {d[key]} ligger i dictionary')
else:
    print(f'Ingen regel med nøkkel {key} i dictionary')
```

Kjøreresultat:

```
Regelen Berkeley --> US ligger i dictionary
```

Eksempel

```
d = {'Berkeley': 'US', 'Cambridge': 'UK'}
key = 'Berkeley'
value = d.get(key)
if value != None:
    print(f'Regelen {key} --> {d[key]} ligger i dictionary')
else:
    print(f'Ingen regel med nøkkel {key} i dictionary')
```

Eksempel

```
d = {'Berkeley': 'US', 'Cambridge': 'UK'}
key = 'Berkeley'
value = d.get(key)
if value != None:
    print(f'Regelen {key} --> {d[key]} ligger i dictionary')
else:
    print(f'Ingen regel med nøkkel {key} i dictionary')
```

Kjøreresultat:

```
Regelen Berkeley --> US ligger i dictionary
```

Å løpe gjennom elementene i en dictionary

Løpe gjennom elementene i vilkårlig rekkefølge:

```
d = {-2:6, 6:3, 3:7}
for key in d:
    print(f'Key = {key} and value = {d[key]}')
```

Løpe gjennom elementene i sortert nøkkelrekkefølge:

```
d = {-2:6, 6:3, 3:7}
for key in sorted(d):
    print(f'Key = {key} and value = {d[key]}')
```

Lage liste over alle nøkler/verdier

```
d = {'Paris': 17.5, 'London': 15.4, 'Madrid': 26.0}

> d.keys()
dict_keys(['Paris', 'London', 'Madrid'])

> list(d.keys())
['Paris', 'London', 'Madrid']

> d.values()
dict_values([17.5, 15.4, 26.0])

> list(d.values())
[17.5, 15.4, 26.0]
```

Eksempel A: Lese to-kolonne fil til dictionary

Datafil:

```
MB.0000                0.00096
MB.0002                0.24787
MB.0005                0.2779
MB.0006                0.29428
MB.0010                0.61225
...    ...
```

Datafil:

```
MB.0000                    0.00096
MB.0002                    0.24787
MB.0005                    0.2779
MB.0006                    0.29428
MB.0010                    0.61225
...    ...
```

Første oppgave: å bestemme hva som skal være nøkkel og hva som skal være verdi.

Datafil:

MB.0000	0.00096
MB.0002	0.24787
MB.0005	0.2779
MB.0006	0.29428
MB.0010	0.61225
...	...

Første oppgave: å bestemme hva som skal være nøkkel og hva som skal være verdi.

Nøkkel: første kolonne
Verdi: andre kolonne

Datafil:

```
MB.0000                    0.00096
MB.0002                    0.24787
MB.0005                    0.2779
MB.0006                    0.29428
MB.0010                    0.61225
...    ...
```

```
bloodtest = {}
with open('blood_test.txt', 'r') as infile:
    for line in infile:
        words = line.split()
        bloodtest[words[0]] = float(words[1])

# Skrive ut verdien til nøkkel MB.0005:
print(bloodtest['MB.0005'])          # 0.2779
```

Eksempel B: Lese tre-kolonne fil til dictionary

Datafil:

```
Oslo        21.8        'Norway'
Bergen      17.6        'Norway'
London      18.1        'UK'
Berlin      19          'Germany'
Paris       23          'France'
Rome        26          'Italy'
Helsinki    17.8        'Finland'
```

Datafil:

```
Oslo        21.8    'Norway'
Bergen      17.6    'Norway'
London      18.1    'UK'
Berlin      19      'Germany'
Paris       23      'France'
Rome        26      'Italy'
Helsinki    17.8    'Finland'
```

Første oppgave: å bestemme hva som skal være nøkkel og hva som skal være verdi.

Nøkkel: første kolonne
Verdi: andre og tredje kolonne (som liste/tuppel)

Datafil:

```
Oslo        21.8        'Norway'
Bergen      17.6        'Norway'
London      18.1        'UK'
Berlin      19          'Germany'
Paris       23          'France'
Rome        26          'Italy'
Helsinki    17.8        'Finland'
```

```
data = {}
# Les fil
with open('cityinfo.txt', 'r') as infile:
    for line in infile:
        words = line.split()
        data[words[0]] = [float(words[1]), words[2]]

# Skriv ut informasjonen til nøkkel Paris
print(data['Paris'])      # [23.0, 'France']
print(data['Paris'][0])   # 23.0
print(data['Paris'][1])   # 'France'
```

Eksempel C: Lese fil med navngitte kolonner

Datafilen table.dat:

```
Navn        Alder       Utdanningssted
Anne        10          Eiksmarka
Tom         15          Marienlyst
Vidar       18          Persbråten
Johanne     17          Wang
Silje       16          Eikeli
Per         19          UiO
```

Eksempel C: Lese fil med navngitte kolonner

Datafilen table.dat:

```
Navn        Alder       Utdanningssted
Anne        10          Eiksmarka
Tom         15          Marienlyst
Vidar       18          Persbråten
Johanne     17          Wang
Silje       16          Eikeli
Per         19          UiO
```

```
data = {}
with open('table.dat', 'r') as infile:
    headers = infile.readline().split()
    for i in range(len(headers)):
        data[headers[i]] = []
    for line in infile:
        words = line.split()
        for i in range(len(headers)):
            data[headers[i]].append(words[i])
```

Eksempel C: Lese fil med navngitte kolonner

Datafilen table.dat:

```
Navn        Alder       Utdanningssted
Anne        10          Eiksmarka
Tom         15          Marienlyst
Vidar       18          Persbråten
Johanne     17          Wang
Silje       16          Eikeli
Per         19          UiO
```

```
data = {}
with open('table.dat', 'r') as infile:
    headers = infile.readline().split()
    for i in range(len(headers)):
        data[headers[i]] = []
    for line in infile:
        words = line.split()
        for i in range(len(headers)):
            data[headers[i]].append(words[i])
```

Les overskrifter

Les data

Quiz 1

Hva skrives ut?

Spørsmål A:

```
d = {-2:-1, -1:0, 0:1, 1:2, 2:-2}
print(d[0])
```

Spørsmål B:

```
d = {-2:-1, -1:0, 0:1, 1:2, 2:-2}
print(d[d[0]])
```

Spørsmål C:

```
d = {-2:-1, -1:0, 0:1, 1:2, 2:-2}
print(d[-2]*d[2])
```

Quiz 1

Hva skrives ut?

Spørsmål A:

```
d = {-2:-1, -1:0, 0:1, 1:2, 2:-2}
print(d[0])                  # 1
```

Spørsmål B:

```
d = {-2:-1, -1:0, 0:1, 1:2, 2:-2}
print(d[d[0]])
```

Spørsmål C:

```
d = {-2:-1, -1:0, 0:1, 1:2, 2:-2}
print(d[-2]*d[2])
```

Quiz 1

Hva skrives ut?

Spørsmål A:

```
d = {-2:-1, -1:0, 0:1, 1:2, 2:-2}
print(d[0])                 # 1
```

Spørsmål B:

```
d = {-2:-1, -1:0, 0:1, 1:2, 2:-2}
print(d[d[0]])              # 2
```

Spørsmål C:

```
d = {-2:-1, -1:0, 0:1, 1:2, 2:-2}
print(d[-2]*d[2])
```

Quiz 1

Hva skrives ut?

Spørsmål A:

```
d = {-2:-1, -1:0, 0:1, 1:2, 2:-2}
print(d[0])                 # 1
```

Spørsmål B:

```
d = {-2:-1, -1:0, 0:1, 1:2, 2:-2}
print(d[d[0]])              # 2
```

Spørsmål C:

```
d = {-2:-1, -1:0, 0:1, 1:2, 2:-2}
print(d[-2]*d[2])           # 2
```

Quiz 2

Hva skrives ut?

```
table = {'age':[35,20], 'name':['Anna','Peter']}
for key in table:
   print(f'{key}: {table[key]}')
```

Quiz 2

Hva skrives ut?

```
table = {'age':[35,20], 'name':['Anna','Peter']}
for key in table:
   print(f'{key}: {table[key]}')
```

SVAR:

age: [35, 20]
name: ['Anna', 'Peter']

(rekkefølgen til linjene kan være omvendt)

name: ['Anna', 'Peter']
age: [35, 20]

Quiz 3

Hva skrives ut:

```
table = {'age':[35,20], 'name':['Anna','Peter']}
vals = list(table.values())
print(vals)
print(vals[0])
print(vals[0][0])
```

Quiz 3

Hva skrives ut:

```
table = {'age':[35,20], 'name':['Anna','Peter']}
vals = list(table.values())
print(vals)
print(vals[0])
print(vals[0][0])
```

SVAR:

```
[[35, 20], ['Anna', 'Peter']]    (evt i omvendt rekkeflg)
[35, 20]
35
```

Quiz 4

Hva skrives ut?

```
table = {'age':[35,20], 'name':['Anna','Peter']}
print(f'{table["name"][1]} er {table["age"][1]}')
```

NB: Merk at vi bruker doble anførselstegn når vi ønsker å angi en tekststreng (f.eks. "name") inni en annen tekststreng (f.eks. f-strengen over). Dette glemte jeg på forelesningen i dag (takk til studenten som sa fra!).

Quiz 4

Hva skrives ut?

```
table = {'age':[35,20], 'name':['Anna','Peter']}
print(f'{table["name"][1]} er {table["age"][1]}')
```

SVAR:

Peter er 20

Quiz 5

Hva blir innholdet i oppslagstabellen d?

Spørsmål A:

```
d = {3:5, 6:7}
e = {4:6, 7:8}
d.update(e)
```

Spørsmål B:

```
d = {3:5, 6:7}
e = {4:6, 7:8}
d.update(e)
d.update(e)
```

Spørsmål C:

```
d = {6:100}
e = {6:6, 7:8}
d.update(e)
```

Quiz 5

Hva blir innholdet i oppslagstabellen d?

Spørsmål A:

```
d = {3:5, 6:7}
e = {4:6, 7:8}
d.update(e)
```

{3:5, 4:6, 6:7, 7:8}

Spørsmål B:

```
d = {3:5, 6:7}
e = {4:6, 7:8}
d.update(e)
d.update(e)
```

Spørsmål C:

```
d = {6:100}
e = {6:6, 7:8}
d.update(e)
```

Quiz 5

Hva blir innholdet i oppslagstabellen d?

Spørsmål A:

```
d = {3:5, 6:7}
e = {4:6, 7:8}
d.update(e)
```

{3:5, 4:6, 6:7, 7:8}

Spørsmål B:

```
d = {3:5, 6:7}
e = {4:6, 7:8}
d.update(e)
d.update(e)
```

{3:5, 4:6, 6:7, 7:8}

Spørsmål C:

```
d = {6:100}
e = {6:6, 7:8}
d.update(e)
```

Quiz 5

Hva blir innholdet i oppslagstabellen d?

Spørsmål A:

```
d = {3:5, 6:7}
e = {4:6, 7:8}
d.update(e)
```

{3:5, 4:6, 6:7, 7:8}

Spørsmål B:

```
d = {3:5, 6:7}
e = {4:6, 7:8}
d.update(e)
d.update(e)
```

{3:5, 4:6, 6:7, 7:8}

Spørsmål C:

```
d = {6:100}
e = {6:6, 7:8}
d.update(e)
```

{6:6, 7:8}

Quiz 6

Filen 'teledata.txt' inneholder info om mobilkunder:

Age	Income	Calls	ID
45	720k	46	A001
27	440k	3	A002
17	0k	52	A006
24	60k	18	A014

Kom med forslag til hvordan vi kan:

a) lagre dataene som fire lister

b) lagre dataene som én liste

c) lagre dataene som en dictionary

Quiz 6

Fire lister:

```
age    = [45, 27, 17, 24]
income = ['720k', '440k', '0k', '60k']
calls  = [46, 3, 52, 18]
id     = ['A001', 'A002', 'A006', 'A014']
```

En liste:

```
table = [[45, 27, 17, 24],
         ['720k', '440k', '0k', '60k'],
         [46, 3, 52, 18],
         ['A001', 'A002', 'A006', 'A014']]
```

En dictionary:

```
table = {'Age':[45, 27, 17, 24,
         'Income':['720k', '440k', '0k', '60k'],
         'Calls':[46, 3, 52, 18],
         'ID':['A001', 'A002', 'A006', 'A014']}
```

Nestet dictionary

Vi har allerede sett eksempler på at verdiene i en dictionary kan være lister.

Verdiene kan også være dictionaries:

```
avstand = {'Oslo':{'Bergen':463,'Trondheim':491,'Tromsø':1798},
           'Bergen':{'Oslo':463,'Trondheim':627},
           'Trondheim':{'Oslo':491,'Tromsø':1129}}
```

```
> avstand['Oslo']['Bergen']
463

> avstand['Trondheim']['Tromsø']
1129
```

Plenumsoppgaver

Exercise A.14: Find difference equations for computing sin x

The purpose of this exercise is to derive and implement difference equations for computing a Taylor polynomial approximation to $\sin x$:

$$\sin x \approx S(x;n) = \sum_{j=0}^{n} (-1)^j \frac{x^{2j+1}}{(2j+1)!}. \tag{A.60}$$

To compute $S(x;n)$ efficiently, write the sum as $S(x;n) = \sum_{j=0}^{n} a_j$, and derive a relation between two consecutive terms in the series:

$$a_j = -\frac{x^2}{(2j+1)2j} a_{j-1}. \tag{A.61}$$

Introduce $s_j = S(x; j-1)$ and a_j as the two sequences to compute. We have $s_0 = 0$ and $a_0 = x$.

a) Formulate the two difference equations for s_j and a_j.

Exercise A.14

Taylor approksimasjon til $\sin x$:

$$\sin x \approx \sum_{j=0}^{n} (-1)^j \frac{x^{2j+1}}{(2j+1)!}$$

Exercise A.14

Taylor approksimasjon til $\sin x$:

$$\sin x \approx \sum_{j=0}^{n} (-1)^j \frac{x^{2j+1}}{(2j+1)!}$$

For å regne ut det jte leddet må vi regne ut:

x^{2j+1}

$(2j + 1)!$

$(-1)^j$

... og så multiplisere og dividere

Exercise A.14

Taylor approksimasjon til $\sin x$:

$$\sin x \approx \sum_{j=0}^{n} (-1)^j \frac{x^{2j+1}}{(2j+1)!}$$

For å regne ut det $j$te leddet må vi regne ut:

- x^{2j+1}
- $(2j+1)!$
- $(-1)^j$
- ... og så multiplisere og dividere

Kan vi gjøre det mer effektivt?

Ide: Det er lett å komme fra ledd $j-1$ til ledd j.

Exercise A.14

Ledd $j - 1$:

$$a_{j-1} = (-1)^{j-1} \frac{x^{2(j-1)+1}}{(2(j-1)+1)!} = (-1)^{j-1} \frac{x^{2j-1}}{(2j-1)!}$$

Exercise A.14

Ledd $j-1$:

$$a_{j-1} = (-1)^{j-1} \frac{x^{2(j-1)+1}}{(2(j-1)+1)!} = (-1)^{j-1} \frac{x^{2j-1}}{(2j-1)!}$$

Ledd j:

$$a_j = (-1)^j \frac{x^{2j+1}}{(2j+1)!} = -\frac{x^2}{2j(2j+1)} \cdot a_{j-1}$$

Exercise A.14

Ledd $j-1$:

$$a_{j-1} = (-1)^{j-1} \frac{x^{2(j-1)+1}}{(2(j-1)+1)!} = (-1)^{j-1} \frac{x^{2j-1}}{(2j-1)!}$$

Ledd j:

$$a_j = (-1)^j \frac{x^{2j+1}}{(2j+1)!} = -\frac{x^2}{2j(2j+1)} \cdot a_{j-1}$$

Dermed: hvis vi allerede har regnet ut forrige ledd, trenger vi bare tre multiplikasjoner + en divisjon + et fortegnsskift for å finne neste!

Exercise A.14

Oppsummert har vi nå at

$$\sin x \approx \sum_{j=0}^{n} a_j$$

hvor

$$a_j = -\frac{x^2}{2j(2j+1)} \cdot a_{j-1} \qquad j = 1, 2, 3, \ldots.$$

I tillegg må vi sette initialbetingelsen $a_0 = x$.

Exercise A.14

Vi kan skrive Taylor-approksimasjonen $\sin x \approx \sum_{j=0}^{n} a_j$ på en litt annen måte:

$$\sin x \approx s_{n+1}$$

hvor

$$\begin{aligned} s_{n+1} &= s_n + a_n \\ a_n &= -\frac{x^2}{2n(2n+1)} \cdot a_{n-1} \end{aligned}$$

og $a_0 = x$.

Spørsmål: I hvilken rekkefølge må vi beregne de to formlene for å løse dette settet av to differenslikninger?

b) Implement the system of difference equations in a function `sin_Taylor(x, n)`, which returns s_{n+1} and $|a_{n+1}|$. The latter is the first neglected term in the sum (since $s_{n+1} = \sum_{j=0}^{n} a_j$) and may act as a rough measure of the size of the error in the Taylor polynomial approximation.

c) Verify the implementation by computing the difference equations for $n = 2$ by hand (or in a separate program) and comparing with the output from the `sin_Taylor` function. Automate this comparison in a test function.

d) Make a table or plot of s_n for various x and n values to illustrate that the accuracy of a Taylor polynomial (around $x = 0$) improves as n increases and x decreases.

Exercise 5.14: Plot data in a two-column file

The file `src/plot/xy.dat`[12] contains two columns of numbers, corresponding to x and y coordinates on a curve. The start of the file looks as this:

```
-1.0000      -0.0000
-0.9933      -0.0087
-0.9867      -0.0179
-0.9800      -0.0274
-0.9733      -0.0374
```

Make a program that reads the first column into a list `x` and the second column into a list `y`. Plot the curve. Print out the mean y value as well as the maximum and minimum y values.

Hint Read the file line by line, split each line into words, convert to `float`, and append to `x` and `y`. The computations with `y` are simpler if the list is converted to an array.

Filename: `read_2columns`.