

i **Nytt dokument**

Institutt for datateknologi og informatikk

Eksamensoppgave i TDT4110 - Konte-eksamen - Informasjonsteknologi, grunnkurs

Faglig kontakt under eksamen:

- Børge Haugset Mobiltelefon: 934 20 190

- Terje Rydland Mobiltelefon: 957 73 463

Eksamensdato: 5/8-2019

Eksamenstid (fra-til): 09:00 – 13:00

Hjelpemiddelkode/Tillatte hjelpemidler: D

Annen informasjon:

Oppgavesettet inneholder 3 oppgaver. Det er angitt i prosent hvor mye hver oppgave og hver deloppgave teller ved sensur. Les igjennom hele oppgavesettet før du begynner å løse oppgavene. Disponer tiden godt!

Gjør rimelige antagelser der du mener oppgaveteksten er ufullstendig, skriv kort hva du antar. Svar kort og klart. Er svaret uklart eller lenger enn nødvendig trekker dette ned.

Merk! Studenter finner sensur i Studentweb. Har du spørsmål om din sensur må du kontakte instituttet ditt. Eksamenkontoret vil ikke kunne svare på slike spørsmål.

1 **Oppgave 1 Teori - 25%**

Marker det du mener er det mest riktige alternativet. Svaralternativene kommer i tilfeldig rekkefølge.

Feil svar gir IKKE minuspoeng.

Hva står random i RAM for?

- At det er tilfeldig hvor i minnet informasjonen lagres.
- At datamaskinen henter informasjon fra et tilfeldig sted i minnet.
- At alle adresser kan hentes direkte.
- At det er tilfeldig hva som lagres i minnet.

Hva er det som benyttes i dag som har erstattet vacuum tubes (vakuurrør)?

- Transistorer
- CPU
- RAM
- Assembly

Hvorfor er det ønskelig å bruke SSD fremfor en vanlig harddisk?

- Det er lettere å lagre data permanent på en SSD.
- En SSD øker minnet på grafikkortet slik at spill og lignende flyter bedre.
- I en SSD lagres data i elektroniske kretser. Det er ingen bevegelige deler, og dermed blir disken raskere og mer pålitelig.
- En SSD er ikke så utsatt for strømtopper og tåler derfor mer enn en magnetisk disk.

Hva er et operativsystem?

- Et system som operer med et lavnivåspråk, f.eks. assembly.
- En prosess der kretser blir printet på en chip.
- Et mye brukt grunnleggende lavnivåspråk som tar hånd om regneoperasjoner i en datamaskin.
- Grunnleggende operasjoner som er nødvendig for å kunne bruke maskinen effektivt, men som ikke er bygd inn i maskinvaren.

Hva er pipelining?

- Et uttrykk for det som skjer når man skriver mye data til harddisken samtidig.
- En teknikk der en CPU kan utføre flere instruksjoner parallelt.
- En teknikk som fungerer som en sikker tunnel mellom din maskin og en tjener.
- En teknikk der man sender data mellom de forskjellige delene i maskinen i «pipes».

Hva skjer om samplingsfrekvensen er for treg?

- Nyquist-effekten vil forekomme
- Man får konstant skurring
- Lydbølger kan forekomme mellom samplene, og man kan miste viktige segmenter av lyden
- Man får mer eksakt gjengiving av lyden

Hvilken av følgende komprimeringer er loss-less?

- MPEG
- JPEG
- MP3
- Run-length koding

Hvor mange symboler kan representeres med 6 bit?

- 64
- 63
- 128
- 32

Hvor mange bytes bruker UNICODE i "worst-case"?

- 2
- 4
- 3
- 1

Hva er latency?

- tiden det tar å komprimere data
- tiden en datamaskin bruker på å starte opp
- tiden det tar for informasjon å bli laget eller levert.
- klokkehastigheten til en datamaskin

Hva er unicast?

- En type adressering der et subsett av datamaskinene i nettverket blir identifisert og mottar en kopi av pakken.
- En type adressering der hver datamaskin i nettverket blir identifisert og mottar en kopi av pakken.
- En type adressering som kan betraktes som en form for multicast.
- En type adressering der en enkelt datamaskin i nettverket blir identifisert og mottar pakken.

Hva er en protokoll i nettverks-sammenheng?

- En protokoll beskriver prosedyrer for feilhåndtering og uventede hendelser.
- En protokoll er et sett med regler over hvordan applikasjonslaget kommuniserer med nettverkslaget.
- En protokoll beskriver innholdet i hver suite (familie).
- En protokoll er en samling entiteter i et nettverk.

Hvilken metode av flytkontroll er mest effektiv?

- Sliding window
- Kontrollpakking
- Stop-and-go
- Sequencing

Hva er sant om transportlaget i TCP/IP?

- Transportlaget sørger for at all data blir levert slik den ble sendt; komplett og i riktig rekkefølge.
- Transportlaget består blant annet av spesifikasjoner om nettverksadressering og det maksimale antallet pakker som et nettverk kan støtte.
- Transportlaget inneholder alle spesifikasjoner relatert til radiofrekvenser.
- Transportlaget spesifiserer metoden for å dele en stor internett-pakke opp i flere små pakker for overføring.

Hva er sant om syklisk sjekksum (Cyclic Redundancy Codes)?

- Syklisk sjekksum brukes i høyhastighetsnettverk fordi den godtar en melding med fast lengde og er ekstremt god til feildetektering.
- En syklisk sjekksum beregning kan bli sett på som en matematisk operasjon i en Galois kropp av 2. orden (Galois field of order 2).
- Den avanserte matematikken som må til i syklisk sjekksum, gjør at en syklisk sjekksum-beregning tar for lang tid til å kunne brukes effektivt.
- Single Parity Checking (SPC) detekterer oftere feil enn syklisk sjekksum.

Hva er phishing?

- Å opptre som en kjent nettside (f.eks. nettbank) for å få tak i personlig informasjon som f.eks. aksesskoder, kontonummer, etc.
- Tap av åndsverk eller annen verdifull informasjon
- Å fiske etter personlig informasjon ved å late som om maskinen er under virusangrep. Angriperen lover brukeren at feilen skal rettes opp dersom brukeren først oppgir kontonummeret sitt.
- At uvedkommende tar kontroll over en brukers datamaskin.

Hvordan kan man best sikre at data som kommer fram er identisk med de som ble sendt (sikre dataintegritet)?

- Bruke passord
- Bruke digitale signaturer
- Bruke hashing
- Bruke kryptering

Hvilken hensikt har brannmurer (firewalls)?

- De skal hindre minnelekkasjer i maskinen.
- De er ment til å beskytte nettstedets integriteten.
- De gjør at programtelleren har mulighet til å utføre flere instruksjoner på samme tid.
- De er ment til å beskytte mot DoS(Denial of Service) og DDoS(distributed denial-of-service)

Hvordan fungerer kryptering med privat nøkkel (i motsetning til kryptering med offentlig nøkkel)?

- En offentlig nøkkel brukes både til å kryptere og dekryptere en melding.
- Hver part får en hemmelig og en offentlig nøkkel. En melding kryptert med en offentlig nøkkel, kan kun dekrypteres med den korresponderende private nøkkelen.
- Partene deler en hemmelig nøkkel som brukes både for kryptering og dekryptering.
- Senderens private nøkkel publiseres slik at mottaker kan dekryptere meldingen.

Hva står VPN for?

- Virtual private network
- Virtual protection network
- Virtual policy network
- Virtual packet network

Maks poeng: 25

Useful Python functions and commands

Built-in:

format(numeric_value, format_specifier)

Formats a numeric value into a string according to the format specifier, which is a string that contains special characters specifying how the numeric value should be formatted. Examples of various formatting characters are "f=floating-point, e=scientific notation, %=percentage, d=integer". A number before the formatting character will specify the field width. A number after the character "." will format the number of decimals.

%

Remainder (modulo operator): Divides one number by another and gives the remainder.

len(s)

Return the length (the number of items) of a string, tuple, list, dictionary or other data structure.

int(x)

Convert a string or number to a plain integer.

float(x)

Convert a string or a number to floating point number.

str([object])

Return a string containing a nicely printable representation of an object.

range(stop)

Returns a list with integers from 0, up to but not including stop. `range(3) = [0, 1, 2]`. Often used in combination with for loops: `for i in range(10)`

range([start], stop[, step])

start: Starting number of the sequence.

stop: Generate numbers up to, but not including, this number.

step: Difference between each number in the sequence.

chr(i)

Return a string of one character whose ASCII code is the integer i. For example, `chr(97)` returns the string 'a'. This is the inverse of `ord()`

ord()

Given a string of length one, return an integer representing the Unicode code point of the character when the argument is a unicode object, or the value of the byte when the argument is an 8-bit string. For example, `ord('a')` returns the integer 97.

tuple(iterable)

Accepts something iterable (list, range etc.) and returns the corresponding tuple.

if x in iterable:

Returns True if x is an item in iterable.

for idx, x in enumerate(iterable)

Enters a loop with x being one and one item from iterable. idx is an integer containing the loop number.

try: except: else: finally:

try:

Code to test

except:

If code fails. Many exception types, like IOError for file operations.

Variant: `except Exception as exc` # Let's you print the exception.

else:

Runs if no exception occurs

finally:

Runs regardless of prior code having succeeded or failed.

String operations:

s.isalnum()

Returns true if the string contains only alphabetic letters or digits and is at least one character of length. Returns false otherwise.

s.isalpha()

Returns true if the string contains only alphabetic letters, and is at least one character in length. Returns false otherwise.

s.isdigit()

Returns true if the string contains only numeric digits and is at least one character in length. Returns false otherwise.

s.isspace()

Returns true if the string contains only whitespace characters, and is at least one character in length. Returns false otherwise. (Whitespace characters are spaces, newlines (`\n`), and tabs (`\t`)).

s.ljust(width)

Return the string left justified in a string of length width.

s.rjust(width)

Return the string right justified in a string of length width.

s.join(list)

Returns a string listing all items in the list, separated by s.

s.lower()

Returns a copy of the string with all alphabetic letters converted to lowercase.

s.upper()

Returns a copy of the string with all alphabetic letters converted to uppercase.

s.strip()

Returns a copy of the string with all leading and trailing white space characters (spaces, newlines and tabs) removed.

s.strip(char)

Returns a copy of the string with all instances of char that appear at the beginning and the end of the string removed.

s.split(str)

Returns a list of all the words in the string, using str as the separator (splits on all whitespace if left unspecified).

str.splitlines([keepends])

Return a list of the lines in the string, breaking at line boundaries. This method uses the universal newlines approach to splitting lines. Line breaks are not included in the resulting list unless keepends is given and true.

Python recognizes "`\r`", "`\n`", and "`\r\n`" as line boundaries for 8-bit strings.

s.endswith(substring)

The substring argument is a string. The method returns true if the string ends with substring.

s.startswith(substring)

The substring argument is a string. The method returns true if the string starts with substring.

s.find(substring)

The substring argument is a string. The method returns the lowest index in the string where substring is found. If substring is not found the method returns -1.

s.replace(old, new)

The old and new arguments are both strings. The method returns a copy of the string with all instances of old replaced by new.

*str.format(*args, **kwargs)*

Perform a string formatting operation. The string on which this method is called can contain literal text or replacement fields delimited by braces {}. Each replacement field contains either the numeric index of a positional argument, or the name of a keyword argument. Returns a copy of the string where each replacement field is replaced with the string value of the corresponding argument.

List operations:

s[i:j:k]

Return slice starting at position i extending to position j every k items. Can also be used for strings.

item in s

Determine whether a specified item is contained in a list.

min(list)

Returns the item that has the lowest value in the sequence.

max(list)

Returns the item that has the highest value in the sequence.

s.append(x)

Append new element x to end of s.

s.insert(index,item)

Insert an item into a list at a specified position given by an index.

s.index(item)

Return the index of the first element in the list containing the specified item.

s.pop()

Return last element and remove it from the list.

s.pop(i)

Return element i and remove it from the list.

s.remove(item)

Removes the first element containing the item.

s.reverse()

Reverses the order of the items in a list.

s.sort()

Rearranges the elements of a list so they appear in ascending order.

Dictionary operations:

d.clear()

Clears the contents of a dictionary

d.get(key, default)

Gets the value associated with a specific key. If the key is not found, the method does not raise an exception. Instead, it returns a default value.

d.items()

Returns all the keys in a dictionary and their associated values as a sequence of tuples.

d.keys()

Returns all the keys in a dictionary as a sequence of tuples.

d.pop(key, default)

Returns the value associated with a specific key and removes that key-value pair from the dictionary. If the key is not found, the method returns a default value.

d.popitem()

Returns a randomly selected key-value pair as a tuple from the dictionary and removes that key-value pair from the dictionary.

d.values()

Returns all the values in dictionary as a sequence of tuples.

File operations:

open()

Returns a file object, and is most commonly used with two arguments: `open(filename, mode)`. Mode can be 'r' (read only), 'w' (writing only), 'a' (appending), 'r+' (both reading and writing). Adding 'b' to the read or write attribute lets you use binary mode to save the value of variables to file and load from them. For binary to work you need to import the pickle library.

f.read(size)

Reads data from file and returns it as a string. Size is an optional and if left out the whole file will be read.

f.readline()

Reads a single line from the file (reads until and including a newline character (`\n`) is found), and returns it as a string.

f.readlines()

Reads data from the file and returns it as a list of strings.

f.write(string)

Writes the contents of string to file.

f.writelines(list)

Writes a sequence of strings (typically a list of strings) to file.

f.close()

Close the file and free up any system resources taken up by the open file.

Library operations:

pickle.dump(obj, file)

Write a pickled representation of obj to the open file object file.

pickle.load(file_object)

Read a string from the open file object file and interpret it as a pickle data stream, reconstructing and returning the original object hierarchy.

2 Oppgave 2a (6%)

Hva blir utskriften fra dette Python-programmet:

```
def myst(a, b):  
    if a > b:  
        return a  
    else:  
        a *= -2  
        return myst(a, b)  
  
print(myst(-1,8))
```

Fyll inn ditt svar her:

Maks poeng: 6

3 Oppgave 2b (6%)

Hva blir skrevet ut når dette kjøres?

```
def myst2_bonus(noe):  
    return "".join(noe[::-1])  
  
def myst2(str):  
    tmp = []  
    for c in str:  
        if ord(c) >= ord('a'):  
            tmp.append(chr(ord(c)-1))  
    return myst2_bonus(tmp)  
  
print(myst2('o4f5n6b7t8l9f'))
```

Fyll inn ditt svar her:

Maks poeng: 6

4 Oppgave 2c (6%)

Hva blir resultatet når denne koden kjøres?

```
def myst3(a, b, c):  
    c += 1  
    if not a:  
        return myst3(not a, b, c)  
    if (a and b):  
        return c  
    elif (not a and not b):  
        return myst3(not a, b, c)  
    elif (a and not b):  
        return myst3(not a, not b, c)  
    else:  
        return myst3(a, not b)  
  
print(myst3(False, False, 0))
```

Velg ett alternativ

- 3
- programmet krasjer
- uendelig løkke
- 8
- 5
- 4

Maks poeng: 6

5 Oppgave 2d (6%)

Hva skrives ut når denne koden kjøres:

```
def myst4(x, y):  
    tmp1 = 0  
    for a in x:  
        tmp2 = 0  
        for b in y:  
            if b > a:  
                tmp2 += b-a  
        tmp1 += tmp2  
    return tmp1  
  
print(myst4([0,1,2,3,4,5],[5,4,3,2,1,0]))
```

Velg ett alternativ

- 25
- 25
- 15
- 15
- 35
- 31

Maks poeng: 6

6 Oppgave 2e (5%)

Hva skrives ut når dette programmet kjører?

```
def myst5(hmm):  
    svar = 0  
    for x in range(1, len(hmm)-1):  
        if hmm[x-1] > hmm[x]:  
            svar = svar + int(hmm[x])  
    return svar  
  
print(myst5('19836565890'))
```

Fyll inn ditt svar her:

Maks poeng: 5

7 Oppgave 2f (6%)

Gitt variabelen tekst med innhold som under:

```
tekst = 'abcdefghijklmnopqrstuvwxyz'
```

Her mangler bokstavene g og h etter f. Hvordan kan du sette disse bokstavene inn på rett plass i teksten?

Velg ett alternativ

```
tekst = list(tekst)
tekst[6:6] = 'gh'
tekst = ''.join(tekst)
```

```
tekst[6:7] = 'gh'
```

```
tekst[6:6] = 'gh'
```

```
tekst = list(tekst)
tekst[5:6] = list('gh')
tekst = ''.join(tekst)
```

```
tekst = list(tekst)
tekst[6:7] = ['g','h']
tekst = ''.join(tekst)
```

Maks poeng: 6

Useful Python functions and commands

Built-in:

format(numeric_value, format_specifier)

Formats a numeric value into a string according to the format specifier, which is a string that contains special characters specifying how the numeric value should be formatted. Examples of various formatting characters are "f=floating-point, e=scientific notation, %=percentage, d=integer". A number before the formatting character will specify the field width. A number after the character "." will format the number of decimals.

%

Remainder (modulo operator): Divides one number by another and gives the remainder.

len(s)

Return the length (the number of items) of a string, tuple, list, dictionary or other data structure.

int(x)

Convert a string or number to a plain integer.

float(x)

Convert a string or a number to floating point number.

str([object])

Return a string containing a nicely printable representation of an object.

range(stop)

Returns a list with integers from 0, up to but not including stop. `range(3) = [0, 1, 2]`. Often used in combination with for loops: `for i in range(10)`

range([start], stop[, step])

start: Starting number of the sequence.

stop: Generate numbers up to, but not including, this number.

step: Difference between each number in the sequence.

chr(i)

Return a string of one character whose ASCII code is the integer i. For example, `chr(97)` returns the string 'a'. This is the inverse of `ord()`

ord()

Given a string of length one, return an integer representing the Unicode code point of the character when the argument is a unicode object, or the value of the byte when the argument is an 8-bit string. For example, `ord('a')` returns the integer 97.

tuple(iterable)

Accepts something iterable (list, range etc.) and returns the corresponding tuple.

if x in iterable:

Returns True if x is an item in iterable.

for idx, x in enumerate(iterable)

Enters a loop with x being one and one item from iterable. idx is an integer containing the loop number.

try: except: else: finally:

try:

Code to test

except:

If code fails. Many exception types, like IOError for file operations.

Variant: `except Exception as exc` # Let's you print the exception.

else:

Runs if no exception occurs

finally:

Runs regardless of prior code having succeeded or failed.

String operations:

s.isalnum()

Returns true if the string contains only alphabetic letters or digits and is at least one character of length. Returns false otherwise.

s.isalpha()

Returns true if the string contains only alphabetic letters, and is at least one character in length. Returns false otherwise.

s.isdigit()

Returns true if the string contains only numeric digits and is at least one character in length. Returns false otherwise.

s.isspace()

Returns true if the string contains only whitespace characters, and is at least one character in length. Returns false otherwise. (Whitespace characters are spaces, newlines (`\n`), and tabs (`\t`)).

s.ljust(width)

Return the string left justified in a string of length width.

s.rjust(width)

Return the string right justified in a string of length width.

s.join(list)

Returns a string listing all items in the list, separated by s.

s.lower()

Returns a copy of the string with all alphabetic letters converted to lowercase.

s.upper()

Returns a copy of the string with all alphabetic letters converted to uppercase.

s.strip()

Returns a copy of the string with all leading and trailing white space characters (spaces, newlines and tabs) removed.

s.strip(char)

Returns a copy of the string with all instances of char that appear at the beginning and the end of the string removed.

s.split(str)

Returns a list of all the words in the string, using str as the separator (splits on all whitespace if left unspecified).

str.splitlines([keepends])

Return a list of the lines in the string, breaking at line boundaries. This method uses the universal newlines approach to splitting lines. Line breaks are not included in the resulting list unless keepends is given and true.

Python recognizes "`\r`", "`\n`", and "`\r\n`" as line boundaries for 8-bit strings.

s.endswith(substring)

The substring argument is a string. The method returns true if the string ends with substring.

s.startswith(substring)

The substring argument is a string. The method returns true if the string starts with substring.

s.find(substring)

The substring argument is a string. The method returns the lowest index in the string where substring is found. If substring is not found the method returns -1.

s.replace(old, new)

The old and new arguments are both strings. The method returns a copy of the string with all instances of old replaced by new.

*str.format(*args, **kwargs)*

Perform a string formatting operation. The string on which this method is called can contain literal text or replacement fields delimited by braces {}. Each replacement field contains either the numeric index of a positional argument, or the name of a keyword argument. Returns a copy of the string where each replacement field is replaced with the string value of the corresponding argument.

List operations:

s[i:j:k]

Return slice starting at position i extending to position j every k items. Can also be used for strings.

item in s

Determine whether a specified item is contained in a list.

min(list)

Returns the item that has the lowest value in the sequence.

max(list)

Returns the item that has the highest value in the sequence.

s.append(x)

Append new element x to end of s.

s.insert(index,item)

Insert an item into a list at a specified position given by an index.

s.index(item)

Return the index of the first element in the list containing the specified item.

s.pop()

Return last element and remove it from the list.

s.pop(i)

Return element i and remove it from the list.

s.remove(item)

Removes the first element containing the item.

s.reverse()

Reverses the order of the items in a list.

s.sort()

Rearranges the elements of a list so they appear in ascending order.

Dictionary operations:

d.clear()

Clears the contents of a dictionary

d.get(key, default)

Gets the value associated with a specific key. If the key is not found, the method does not raise an exception. Instead, it returns a default value.

d.items()

Returns all the keys in a dictionary and their associated values as a sequence of tuples.

d.keys()

Returns all the keys in a dictionary as a sequence of tuples.

d.pop(key, default)

Returns the value associated with a specific key and removes that key-value pair from the dictionary. If the key is not found, the method returns a default value.

d.popitem()

Returns a randomly selected key-value pair as a tuple from the dictionary and removes that key-value pair from the dictionary.

d.values()

Returns all the values in dictionary as a sequence of tuples.

File operations:

open()

Returns a file object, and is most commonly used with two arguments: `open(filename, mode)`. Mode can be 'r' (read only), 'w' (writing only), 'a' (appending), 'r+' (both reading and writing). Adding 'b' to the read or write attribute lets you use binary mode to save the value of variables to file and load from them. For binary to work you need to import the pickle library.

f.read(size)

Reads data from file and returns it as a string. Size is an optional and if left out the whole file will be read.

f.readline()

Reads a single line from the file (reads until and including a newline character (`\n`) is found), and returns it as a string.

f.readlines()

Reads data from the file and returns it as a list of strings.

f.write(string)

Writes the contents of string to file.

f.writelines(list)

Writes a sequence of strings (typically a list of strings) to file.

f.close()

Close the file and free up any system resources taken up by the open file.

Library operations:

pickle.dump(obj, file)

Write a pickled representation of obj to the open file object file.

pickle.load(file_object)

Read a string from the open file object file and interpret it as a pickle data stream, reconstructing and returning the original object hierarchy.

i **Beskrivelse**

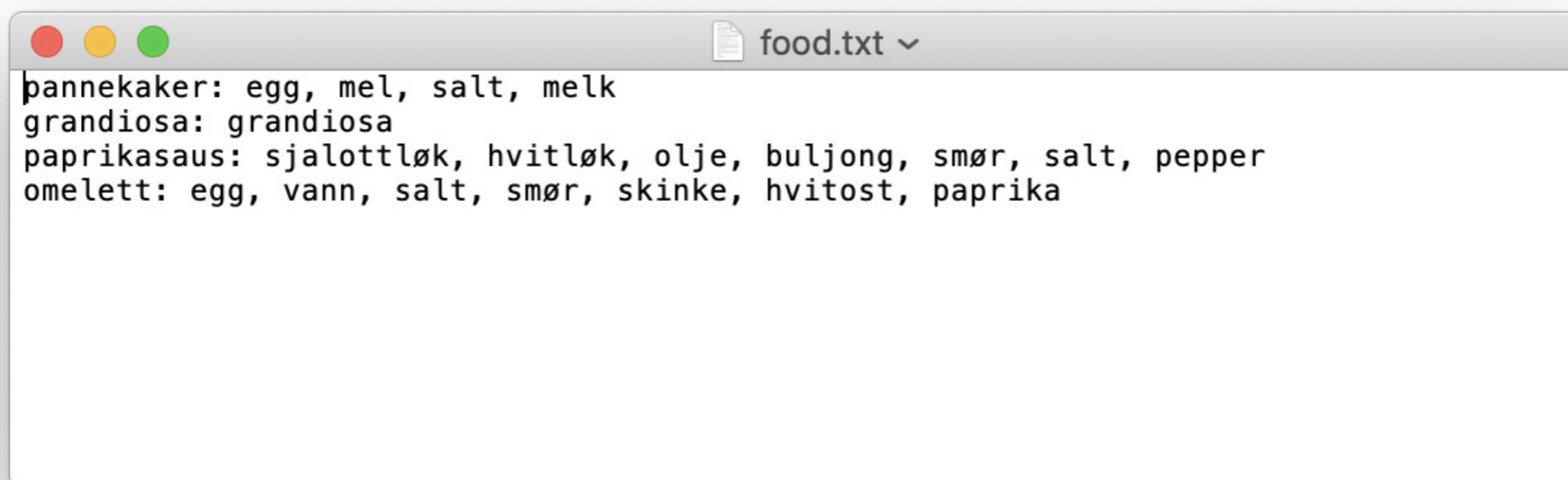
Del 3 – Restauranthjelp

Bakgrunn

Programmeringsoppgaven tar utgangspunkt i oppskrifter. Et sett med ingredienser til oppskrifter skal leses inn fra fil. Deretter skal du lage funksjoner for å finne oppskrifter med gitte ingredienser, og komme med forslag til en tilfeldig rett som inneholder disse.

Et sett med navn på matretter og tilhørende ingredienser befinner seg i filen `food.txt`. Den har følgende struktur: (Det er ikke et ekstra linjeskift mellom hver rett.)

Food.txt:



```
pannekaker: egg, mel, salt, melk
grandiosa: grandiosa
paprikasaus: sjalottløk, hvitløk, olje, buljong, smør, salt, pepper
omelett: egg, vann, salt, smør, skinke, hvitost, paprika
```

8 Oppgave 3-1 (5%)

Oppgave 3-1 Lesing fra fil: `read_file(filename)`:

Skriv funksjonen `read_file(filename)` som skal lese inn en fil, og returnere en liste. Hver linje i filen blir ett element i denne listen, slik som i eksempelet under:

```
>>> print(read_file('food.txt'))  
['pannekaker: egg, mel, salt, melk\n', 'grandiosa:  
grandiosa\n', 'paprikasaus: sjalottløk, hvitløk, olje,  
buljong, smør, salt, pepper\n', 'omelett: egg, vann, salt,  
smør, skinke, hvitost, paprika\n']
```

Skriv ditt svar her...

1	
---	--

Maks poeng: 5

9 Oppgave 3-2 (5%)

Oppgave 2 Lag et sett med ingredienser: **fix_ingredients(string)**

Gitt en streng med ord atskilt av komma og mellomrom ' ', lag funksjonen **fix_ingredients(string)** som returnerer en liste med alle substrenger som elementer i en liste. Funksjonen skal fungere på vilkårlige matvarer, som vist i eksempelet under. Legg merke til hva som skjer med mellomrom og andre 'whitespace'-tegn.

```
>>> print(fix_ingredients('pannekaker: egg, mel, salt, melk\n'))  
['pannekaker: egg', 'mel', 'salt', 'melk']
```

Skriv ditt svar her...

1	
---	--

Maks poeng: 5

10 Oppgave 3-3 (5%)

Oppgave 3: gjør om liste til dictionary: **make_dict(foodlist)**:

Du skal nå lage en funksjon **make_dict(foodlist)**. Denne tar inn resultatet av oppgave 3-1, altså en liste med strenger. Funksjonen skal returnere en ordbok (dictionary). Nøkkelen til hvert oppslag skal være alt som står foran det første kolonet ':' i hvert element (altså matretten det er snakk om). Innholdet bak nøkkelen skal være en liste som inneholder ingrediensene til hver matrett. Funksjonen skal fungere for vilkårlige matretter, og som vist under:

```
>>> mat = read_file('food.txt')
>>> food_dict = make_dict(mat)
>>> print(food_dict)

{'pannekaker': ['egg', 'mel', 'salt', 'melk'],
 'grandiosa': ['grandiosa'],
 'paprिकासaus': ['sjalottløk', 'hvitløk', 'olje', 'buljong', 'smør', 'salt', 'pepper'],
 'omelett': ['egg', 'vann', 'salt', 'smør', 'skinke', 'hvitost', 'paprika']}
```

Skriv ditt svar her...

1	
---	--

Maks poeng: 5

11 Oppgave 3-4 (10%)

Oppgave 4: Skriv oppsummering og ingredienser for én spesifikk rett: **print_recipe()**:

Nå skal du *bruke ordboken* `food_dict` fra forrige oppgave. Funksjonen **`print_recipe()`** skal ta inn denne ordboken samt en matrett (streng), og skrive ut informasjon om hvor mange ingredienser retten inneholder og hva disse er. Under vises hva funksjonen har som input, og hva som skal skrives ut. Legg merke til at listen over ingredienser skal skrives ut på en fin måte, f.eks som vist under:

```
>>> food_dict = make_dict(read_file('food.txt'))
>>> print_recipe(food_dict, 'pannekaker')

pannekaker has 4 ingredients: egg, mel, salt, melk

>>> print_recipe(food_dict, 'sylte')

No dish called sylte
```

Skriv ditt svar her...

1	
---	--

Maks poeng: 10

12 **Oppgave 3-5 (5%)**

Oppgave 5: Sorter retter etter ingredienser: **all_recipes_with(food)**:

Nå skal du søke i den samme ordboken som i forrige oppgave. Du skal lage en ny ordbok, der nøkkelen er hver type ingrediens. Verdien som er knyttet til hver nøkkel er en liste som inneholder alle matrettene som inneholder akkurat denne ingrediensen. Funksjonen skal returnere den nye ordboken, se eksempel under:

```
>>> recipe_dishes = all_recipes_with(food_dict)

>>> recipe_dishes

{'sjalottløk': ['paprikasaus'],
 'egg': ['pannekaker', 'omelett'],
 'skinke': ['omelett'],
 'buljong': ['paprikasaus'],
 'olje': ['paprikasaus'],
 'pepper': ['paprikasaus'],
 'salt': ['pannekaker', 'paprikasaus', 'omelett'],
 'melk': ['pannekaker'],
 'smør': ['paprikasaus', 'omelett'],
 'vann': ['omelett'],
 'paprika': ['omelett'],
 'mel': ['pannekaker'],
 'hvitløk': ['paprikasaus'],
 'hvitost': ['omelett'],
 'grandiosa': ['grandiosa']}
```

Skriv ditt svar her...

1	
---	--

Maks poeng: 5

13 Oppgave 3-6 (5%)

Oppgave 6: Velg en tilfeldig matrett med en ingrediens og skriv den ut på en fin måte

Til nå har du laget to ordbøker som inneholder informasjon om matrettene i tekstfilen. Du skal nå skrive kode som skriver ut antall ingredienser og innholdet til en tilfeldig matrett som inneholder egg. Du har et sett med funksjoner laget over til å hjelpe deg med arbeidet, i tillegg til de to ordbøkene som er beskrevet i tidligere deloppgaver (*food_dict* og *recipe_dishes*). Du må selv lage eventuelle hjelpefunksjoner du trenger ut over dette, men i utgangspunktet trenger du ikke lage noen nye funksjoner for å gjennomføre denne oppgaven. Du trenger ikke skrive at du importerer moduler dersom det er lett å forstå hva du bruker. Du skal som sagt basere koden din på at du skal vise innholdet til en tilfeldig rett som inneholder egg, noe teksten under baserer seg på:

```
>>> food_dict = make_dict(read_file('food.txt'))
>>> recipe_dishes = all_recipes_with(food_dict)

>>> # Do stuff here that involves the above mentioned variables, all
>>> # functions, and the ingredient 'egg' will end up writing:
Today you'll be eating pannekaker
pannekaker has 4 ingredients: egg, mel, salt, melk

But it might as well write:

Today you'll be eating omelett
omelett has 6 ingredients: egg, salt, smør, skinke, hvitost,
paprika
```

Skriv ditt svar her...

1	
---	--

Maks poeng: 5

14 Oppgave 3-7 (5%)

```
>>> value_food(food_dict)
Den dyreste retten er paprikasaus som koster 94
```

Oppgave 7: Beregn bokstavverdien til en matrett: **value_food(food)**:

Gitt at vokaler (æiouyæøå) har verdien 5, og konsonanter har verdien 1.

Skriv funksjonen **value_food(dict)**. Denne skal ta inn en ordbok med matretter slik som `food_dict` (forklart i oppgave 3-3 / 10). Den skal skrive ut den matretten som har ingredienser med totalt sett høyest bokstavverdi, og verdien av denne matretten.

NB: Du må gjerne lage ekstra funksjoner som kan hjelpe deg på veien, alt må ikke gjøres spesifikt i `value_food`.

Skriv ditt svar her...

1	
---	--

Maks poeng: 5