

i Kopi av Forsideinformasjon

Institutt for datateknologi og informatikk

Eksamensoppgave i TDT4110 - Informasjonsteknologi, grunnkurs

Faglig kontakt under eksamen:

- Børge Haugset Mobiltelefon: 934 20 190

- Yngve Dahl Mobiltelefon: 905 27 892

Eksamensdato: 7. august 2018

Eksamenstid (fra-til): 09:00 – 13:00

Hjelpemiddelkode/Tillatte hjelpemidler: D

Annen informasjon:

Oppgavesettet inneholder 4 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. Eksamenskontoret vil ikke kunne svare på slike spørsmål.

1 Kopi av Flervalgsoppgave (25%)

Kryss av det du mener er det riktige svaret. Kun ett svar er helt riktig. For hvert spørsmål gir korrekt avkryssing 1 poeng. Feil avkryssing gir $-1/2$ (minus et halvt) poeng. Blankt svar gir 0 poeng. Du får ikke mindre enn 0 poeng totalt på denne oppgaven. Der det er spesielle uttrykk står den engelske oversettelsen i parentes.

1. Et lokalt nettverk (LAN) består av 6 datamaskiner. Nettverket er organisert som en mesh-topologi. Hvor mange forbindelser er det mellom datamaskinene i nettverket?

- ☐ 12
- ☐ 15
- ☐ 18
- ☐ 36

2. Hva blir binærrepresentasjonen av tallverdien 345?

- ☐ 101100100
- ☐ 101011001
- ☐ 1001000101
- ☐ 11001110

3. Et tegn i en tekst blir i en datamaskin representert som...

- ☐ En indeks inni en tegntabell
- ☐ En grafisk bitmap av tegnet
- ☐ En verdi i datamaskinens tekstminne
- ☐ Et visst antall qubits i en lossy overføringsprotokoll

4. Et bilde har 1920x1200 piksler, i 16 bit fargeformat. Hvor mye plass trenger det ukomprimert?

- ☐ Omtrent 3 MB (megabyte)
- ☐ Omtrent 4.5 MB
- ☐ Omtrent 6 MB
- ☐ Omtrent 6 KB (kilobyte)

5. Hvor mange ulike verdier kan representeres med 12 bit?

- ☐ 2048
- ☐ 1024
- ☐ 1536
- ☐ 4096

6. Hvilket utsagn omkring kompilering og tolking er korrekt

- ☐ Ved tolking oversettes et program til et annet programmeringsspråk
- ☐ Ved tolking oversettes kodelinjene en for en og utføres med en gang
- ☐ Ved kompilering oversettes kodelinjene en for en og utføres med en gang
- ☐ Utførelse av tolket kode er alltid raskere enn utførelse av kompilert kode

7. Hvilken hensikt har en digital signatur?

- ☐ Garantere at en melding kun kan leses av en bestemt mottaker
- ☐ Garantere at en melding kommer fra en bestemt avsender.
- ☐ Både alternativ a og b
- ☐ Ingen av alternativene over

8. Hvilken av påstandene om kryptering er GAL?

- ☐ En melding kryptert med en avsenders private nøkkel kan kun dekrypteres med avsenders offentlige nøkkel.
- ☐ En melding kryptert med en mottakers offentlige nøkkel kan kun dekrypteres med mottakers private nøkkel.
- ☐ En melding kryptert med en avsenders private nøkkel kan dekrypteres med hvilken som helst offentlig nøkkel.
- ☐ Det finnes systemer som kun benytter private nøkler for kryptering og dekryptering.

9. Hvilket formål tjener en brannmur (firewall) i et datanettverk?

- ☐ Sikre at rutere i datanettverket ikke blir overopphet i situasjoner med stor datatrafikk.
- ☐ Kontroll av brukernavn og passord i forbindelse med pålogging på nettverket.
- ☐ Beskytte et datanettverk ved å undersøke pakkehodet til datapakker på vei inn eller ut av nettverket.
- ☐ Beskytte et datanettverk ved å undersøke både pakkehodet til og pakkeinnholdet av datapakker på vei inn og ut av nettverket.

10. Hva er gjennomsnittlig kompleksitet til sortering hvis du bruker innsetting (insertion sort)?

- ☐ $\Theta(n^2)$
- ☐ $\Theta(n \log n)$
- ☐ $\Theta(n)$
- ☐ $\Theta(2n)$

11. For et problem av størrelsen n finnes fire algoritmer med forskjellig tids - kompleksitet. Hvilken vil bruke lengst tid (gjennomsnittlig) på store problemer?

- ☐ $\Theta(n^2)$
- ☐ $\Theta(2^n)$
- ☐ $\Theta(\log n)$
- ☐ $\Theta(n)$

12. Hvilken av disse lagringsenhetene er IKKE en sekundærlagringsenhet?

- ☐ Hurtigbufferet (cache) i datamaskinen.
- ☐ En SSD satt rett i PCI Expressbussen
- ☐ En minnepinne
- ☐ En CD-brenner

13. En device driver er:

- ☐ En spesialdatamaskin for kjøretøy, som er god til visuell analyse.
- ☐ Enheten som holder rede på neste instruksjon som skal utføres av en prosessor.
- ☐ Spesialisert programvare for input/output, slik at utstyr kan kommunisere med resten av systemet.
- ☐ Ingen av alternativene er riktig

14. Ordstørrelse (word size) for en prosessor er:

- ☐ Antall ord i en tekst som kan sammenlignes i et søk.
- ☐ Antall bokstaver som kan behandles i en tekststreng.
- ☐ Antall bit en prosessor maksimalt kan prosessere på en gang.
- ☐ Ingen av alternativene er riktig.

15. Hva sier Nyquist-regelen om samplingsfrekvensen?

- ☐ Samplingsfrekvensen må være minst dobbelt så rask som den høyeste lydfrekvensen.
- ☐ Samplingsfrekvensen må være minst halvparten av den høyeste lydfrekvensen.
- ☐ Samplingsfrekvensen må være minst den samme som den høyeste lydfrekvensen.
- ☐ Samplingsfrekvensen må alltid være 48KHz

16. Hva vil det si at en datamaskin er deterministisk?

- ☐ Den har en pessimistisk livsanskuelse som avviser fri vilje.
- ☐ Når den skal velge hvilken instruksjon den skal behandle neste gang har den ikke noe valg, men baserer valget på programmet og dataene den gis.
- ☐ Når datamaskinen velger hva den skal gjøre så velger den hva som er den rette instruksjonen å kalle nå basert på programmert intuisjon.
- ☐ Ingen av svarene er riktige.

17. Hvilken farge beskrives ved den heksadesimale koden «#fff00» i RGB?

- ☐ Grønn
- ☐ Rød
- ☐ Gul
- ☐ Blå

18. Når er et sekvensialt søk den mest effektive fremgangsmåten?

- ☐ Når dataene er tekst (ikke tall) i usortert rekkefølge.
- ☐ Når dataene er tall i sortert rekkefølge.
- ☐ Når dataene ligger enten i slutten eller i starten av listen.
- ☐ Når dataene ligger i starten av listen.

19. Hvilket av de følgende utsagn er sant om forskjellen på IP versjon 4 og IP versjon 6?

- ☐ Begge bruker like mange bit, men IPv6 bruker en komprimeringsalgoritme for å få plass til flere enheter.
- ☐ IPv6 ble mye diskutert men aldri satt aktivt ut i drift.
- ☐ IPv6 bruker færre bit per navneadresse, men får plass til like mange.
- ☐ IPv6 bruker dobbelt så mange bit per adresse som den gamle.

20. Hva gjør en navnetjener (DNS)?

- ☐ Den oversetter domenenavn til IP-adresser.
- ☐ Den holder rede på hvilke kjørende prosesser på datamaskinen som snakker med hvilke andre datamaskiner, og styrer datapakker fra nettverket rett.
- ☐ Den oversetter IP-adresser til domenenavn.
- ☐ Ingen av alternativene over er riktige.

Maks poeng: 20

2 Kopi av Ny oppgave

For hvert spørsmål gir korrekt avkryssing 1 poeng. Feil avkryssing eller mer enn ett kryss gir $-1/2$ poeng. Blankt svar gir 0 poeng. Du får ikke mindre enn 0 poeng totalt på denne oppgaven. Der det er spesielle uttrykk står den engelske oversettelsen i parentes.

Det er ett alternativ som er korrekt for hvert spørsmål. For å velge et nytt alternativ eller angi blankt svar må du først trykke på det du har valgt for å fjerne valget.

1. Et lokalt nettverk (LAN) består av 6 datamaskiner. Nettverket er organisert som en mesh-topologi. Hvor mange forbindelser er det mellom datamaskinene i nettverket?

☐ 12

☐ 15

☐ 18

☐ 36

2. Hva blir binærrepresentasjonen av tallverdien 345?

☐ 101100100

☐ 101100100

☐ 1001000101

☐ 11001110

3. Et tegn i en tekst blir i en datamaskin representert som...

☐ En indeks inni en tegntabell

☐ En grafisk bitmap av tegnet

☐ En verdi i datamaskinens tekstminne

☐ Et visst antall qubits i en lossy overføringsprotokoll

4. Et bilde har 1920x1200 piksler, i 16 bit fargeformat. Hvor mye plass trenger det ukomprimert?

- ☐ Omtrent 3 MB (megabyte)
- ☐ Omtrent 4.5 MB
- ☐ Omtrent 6 MB
- ☐ Omtrent 6 KB (kilobyte)

5. Hvor mange ulike verdier kan representeres med 12 bit

- ☐ 2048
- ☐ 1024
- ☐ 1636
- ☐ 4096

6. Hvilket utsagn omkring kompilering og tolking er korrekt

- ☐ Ved tolking oversettes et program til et annet programmeringsspråk
- ☐ Ved tolking oversettes kodelinjene en for en og utføres med en gang
- ☐ Ved kompilering oversettes kodelinjene en for en og utføres med en gang
- ☐ Utførelse av tolket kode er alltid raskere enn utførelse av kompilert kode

7. Hvilken hensikt har en digital signatur?

- ☐ Garantere at en melding kun kan leses av en bestemt mottaker
- ☐ Garantere at en melding kommer fra en bestemt avsender
- ☐ Både alternativ a og b
- ☐ Ingen av alternativene over

8. Hvilken av påstandene om kryptering er GAL?

- ☐ En melding kryptert med en avsenders private nøkkel kan kun dekrypteres med avsenders offentlige nøkkel
- ☐ En melding kryptert med en mottakers offentlige nøkkel kan kun dekrypteres med mottakers private nøkkel
- ☐ En melding kryptert med en avsenders private nøkkel kan dekrypteres med hvilken som helst offentlig nøkkel
- ☐ Det finnes systemer som kun benytter private nøkler for kryptering og dekryptering

9. Hvilket formål tjener en brannmur (firewall) i et datanettverk?

- ☐ Sikre at rutere i datanettverket ikke blir overopphet i situasjoner med stor datatrafikk
- ☐ Kontroll av brukernavn og passord i forbindelse med pålogging på nettverket
- ☐ Beskytte et datanettverk ved å undersøke pakkehodet til datapakker på vei inn eller ut av nettverket
- ☐ Beskytte et datanettverk ved å undersøke både pakkehodet til og pakkeinnholdet av datapakker på vei inn og ut av nettverket

10. Hva er gjennomsnittlig kompleksitet til sortering hvis du bruker innsetting (insertion sort)?

- ☐ $\Theta(n^2)$
- ☐ $\Theta(n \log n)$
- ☐ $\Theta(n)$
- ☐ $\Theta(2n)$

11. For et problem av størrelsen n finnes fire algoritmer med forskjellig tids - kompleksitet. Hvilken vil bruke lengst tid (gjennomsnittlig) på store problemer?

- ☐ $\Theta(n^2)$
- ☐ $\Theta(2^n)$
- ☐ $\Theta(n)$
- ☐ $\Theta(\log n)$

12. Hvilken av disse lagringsenhetene er IKKE en sekundærlagringsenhet?

- ☐ Hurtigbufferet (cache) i datamaskinen
- ☐ En SSD satt rett i PCI Expressbussen
- ☐ En minnepinne
- ☐ En CD-brenner

13. En device driver er:

- ☐ En spesialdatamaskin for kjøretøy, som er god til visuell analyse
- ☐ Enheten som holder rede på neste instruksjon som skal utføres av en prosessor
- ☐ Spesialisert programvare for input/output, slik at utstyr kan kommunisere med resten av systemet
- ☐ Ingen av alternativene er riktig

14. Ordstørrelse (word size) for en prosessor er:

- ☐ Antall ord i en tekst som kan sammenlignes i et søk
- ☐ Antall bokstaver som kan behandles i en tekststreng
- ☐ Antall bit en prosessor maksimalt kan prosessere på en gang
- ☐ Ingen av alternativene er riktige

15. Hva sier Nyquist-regelen om samplingsfrekvensen?

- ☐ Samplingsfrekvensen må være minst dobbelt så rask som den høyeste lydfrekvensen
- ☐ Samplingsfrekvensen må være minst halvparten av den høyeste lydfrekvensen
- ☐ Samplingsfrekvensen må være minst den samme som den høyeste lydfrekvensen
- ☐ Samplingsfrekvensen må alltid være 48KHz

16. Hva vil det si at en datamaskin er deterministisk?

- ☐ Den har en pessimistisk livsanskuelse som avviser fri vilje
- ☐ Når den skal velge hvilken instruksjon den skal behandle neste gang har den ikke noe valg, men baserer valget på programmet og dataene den gis
- ☐ Når datamaskinen velger hva den skal gjøre så velger den hva som er den rette instruksjonen å kalle nå basert på programmert intuisjon
- ☐ Ingen av svarene er riktige

17. Hvilken farge beskrives ved den heksadesimale koden «#ffff00» i RGB?

- ☐ Grønn
- ☐ Rød
- ☐ Gul
- ☐ Blå

18. Når er et sekvensialt søk den mest effektive fremgangsmåten?

- ☐ Når dataene er tekst (ikke tall) i usortert rekkefølge
- ☐ Når dataene er tall i sortert rekkefølge
- ☐ Når dataene ligger enten i slutten eller i starten av listen
- ☐ Når dataene ligger i starten av listen

19. Hvilket av de følgende utsagn er sant om forskjellen på IP versjon 4 og IP versjon 6?

- ☐ Begge bruker like mange bit, men IPv6 bruker en komprimeringsalgoritme for å få plass til flere adresser
- ☐ IPv6 ble mye diskutert men aldri satt aktivt ut i drift
- ☐ IPv6 bruker færre bit per navneadresse sammenliknet med IPv4, men får plass til like mange adresser
- ☐ IPv6 bruker dobbelt så mange bit per adresse som IPv4.

20. Hva gjør en navnetjener (DNS)?

- ☐ Den oversetter domenenavn til IP-adresser
- ☐ Den holder rede på hvilke kjørende prosesser på datamaskinen som snakker med hvilke andre datamaskiner, og styrer datapakker fra nettverket rett
- ☐ Den oversetter IP-adresser til domenenavn
- ☐ Ingen av alternativene over er riktige

Maks poeng: 20

3 Kopi av Oppgave 2a (5%)

Hva blir skrevet ut til skjerm når du kjører programmet vist under? (5%)

```
def myst1(x, y, z):  
    t = x  
    x = y  
    y = t  
    for i in range(z):  
        z = x + y  
        y *= 2  
    return z  
  
print(myst1(2, 4, 6))
```

Svar: (3%)

Beskriv med en setning hva funksjonen gjør (2%)

Maks poeng: 10

4 Kopi av Oppgave 2b (5%)

Hva blir skrevet ut til skjerm når du kjører programmet vist under?

```
def myst2(x, y):  
    s = ""  
    for i in range(y-1, len(x), y):  
        s+= x[i]  
    return s  
  
s = "dsigbtmsgmbknrenvrpqkxcunvltbttmx"  
print(myst2(s, 3))
```

Svar: (3%)

Beskriv med en setning hva funksjonen gjør (2%)

Maks poeng: 10

5 Kopi av Oppgave 2c (5%)

Hva blir skrevet ut til skjerm når du kjører programmet vist under?

```
def myst3(a):  
    s = ''  
    for r in a:  
        for c in r:  
            s+=chr(ord('A')+c)  
    return s  
  
z = ((2,0,11,8,5),(14,17,13,8,0))  
print(myst3(z))
```

Svar; (3%)

Beskriv med en setning hva funksjonen gjør (2%)

Maks poeng: 10

6 Kopi av Oppgave 2d (5%)

Hva blir skrevet ut til skjerm når du kjører programmet vist under?

```
def myst4(a,b,c):  
    s = []  
    for i in range(len(a)):  
        s.append(a[i]+b[i]+c[i])  
    s = tuple(s)  
    return s
```

```
x = (1,2,3,4,5,6)  
y = (2,4,6,8,10,12)  
z = (3,6,9,12,15,18)  
print(myst4(x,y,z))
```

Svar:

Beskriv med en setning hva funksjonen gjør (2%)

Maks poeng: 10

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 Error:

A *try clause* lets you execute code between the *try:* and *except* keywords. If an exception occurs, the code following the *except* keyword is executed. One important Error type is `IOError`, which is raised if an I/O operation (such as opening a file on disk) fails.

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 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 in k steps. 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).

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 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.close()

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

i **Kopi av PasswordManager**

Du kan anta at alle funksjonene mottar gyldige argumenter (inn-verdier), og at filen alltid lar seg åpne. Du kan benytte deg av funksjoner fra andre deloppgaver selv om du ikke har løst de deloppgavene.

Du sitter på eksamen i pythonprogrammering. Du blir bedt om å lage et program for å håndtere passord som brukes på nettsteder. Her skal du legge inn passord for ulike nettsteder du vil huske, liste dem opp, og endre passord hvis du må. Du skal til og med kunne sjekke om du bruker det samme passordet på ulike steder.

Passord blir i dette systemet lagret i en dictionary. Hvert innslag i denne dictionary er altså knyttet til ett brukernavn og en liste over passord til nettstedet:

```
notebook = {'Nettside': ['brukernavn', ['gammeltpassord', 'nyttpassord']]}
```

I eksempelet over er det vist to passord, ett nytt og ett gammelt. Det kan være fra ett til et vilkårlig antall passord lagret, alt etter hvor mange ganger det er endret for nettstedet.

Vi vil bruke variabelen notebook til denne passordlisten i hele denne oppgaven. Du trenger ikke sjekke for alle typer feil i inputdata dersom det ikke er spesifisert.

7 Kopi av a) Legg til en nettside: (5%)

Lag funksjonen `addSite(notebook)`.

Et nytt nettsted defineres ved et nettstedsnavn, et brukernavn, og et passord. Du skal lage en funksjon som har en dictionary kalt 'notebook' som spesifisert over som input. Funksjonen skal returnere notebook, der det er lagt til et nytt nettsted med et brukernavn og et passord. Dersom nettstedet allerede er lagt inn skal funksjonen gi beskjed om dette, og returnere notebook uten å ha endret den.

Eksempel på kall av funksjon (bruker-input er skrevet med fet font):

```
>>> notebook = {}
>>> notebook = addSite(notebook)
What is the site: reddit
Write username: urgle
Write password: asdf1234
Account added for reddit.
>>> print(notebook)
{'reddit': ['urgle', ['asdf1234']]}
```

Skriv ditt svar her...

1

Maks poeng: 10

8 Kopi av b) Skriv ut en liste over nettsider og deres nyeste passord: (5%)

Lag funksjonen `showSites(notebook)`.

Funksjonen tar inn den samme notebook-dictionary, og skriver ut en oversikt over alle nettsteder, brukernavn og det **siste** passordet som er lagt inn (sist i listen). Hvis nettstedet har et navn som er mer enn 15 tegn langt skal du bare bruke de 15 første, mens selve feltet for nettsted skal være 17 langt. For brukernavnet holder det å bruke 15 tegn.

Eksempel: (input splittet på flere linjer for å få det mer leselig)

```
>>> notebook =
{'Facebook': ['Urgle', ['pwd1']],
 'reddit': ['urgle', ['asdf1234', 'pwd1']],
 'Aftenposten': ['ivrig', ['pwdold', 'pwdnew']] }
>>> showSites(notebook)
Nettsted:      Brukernavn:      Passord:
Facebook:      Urgle           pwd1
reddit:        urgle           pwd1
Aftenposten:   ivrig           pwdnew
```

Skriv ditt svar her...

Maks poeng: 10

9 Kopi av c) Hjelpefunksjon: formattering av liste til streng (3%)

Lag funksjonen `formatList(list)`.

Du vil snart ha bruk for å formatere en liste full av strenger til en mer lesbar tekst. Funksjonen tar imot en liste, og returnerer en streng med alle elementene. Hvert element skal være separert av komma og mellomrom `' , '`.

Eksempel:

```
>>> tmp = formatList(['pwd', 'pwd1', 'pwd2'])
>>> print(tmp)
pwd, pwd1, pwd2
```

Skriv ditt svar her...

1	
---	--

Maks poeng: 10

10 Kopi av d) Endre passord på nettsted: (6%)

Lag funksjonen `editSite(notebook, site)`.

På grunn av sikkerheten bør en endre passordet på nettsteder innimellom. Funksjonen `editSite` tar inn `notebook`, samt navnet på nettstedet vi vil legge et nytt passord til for. Vi ønsker å lagre de gamle passordene. Hvis det nye passordet allerede har blitt brukt skal den liste opp passordene som er brukt for dette nettstedet før, og spørre etter et nytt. Hvis det nye passordet ikke er brukt før skal det legges til i slutten av passordlisten for dette nettstedet. Bruk funksjonen fra c) for å hjelpe deg. Funksjonen `editSite` skal returnere en oppdatert `notebook`.

Eksempel på kall av funksjon (bruker-input er skrevet med **fet font**):

```
>>> notebook = {'Facebook': ['Urgle', ['pwd1', 'pwd2']]}
>>> editSite(notebook, 'Facebook')
Add new site password for Facebook: pwd2
'pwd2' has been used for Facebook already.
The following passwords have been used: pwd1, pwd2
Add new site password for Facebook: pwd3
Facebook has been updated with a new password.
>>> print(notebook)
{'Facebook': ['Urgle', ['pwd1', 'pwd2', 'pwd3']]}
```

Skriv ditt svar her...

Maks poeng: 10

11 Kopi av e) Sjekke om passord er brukt flere steder: (6%)

Lag funksjonen `secureSites(notebook)`.

Det er viktig å ikke bruke det samme passordet på flere steder. Denne funksjonen skal ta inn `notebook`, og gå igjennom hvert eneste av de sist innlagte passordene for hvert nettsted. Hvis et passord brukes på mer enn ett nettsted skal den vise hvilket passord dette gjelder, og hvilke nettsteder det er brukt på. Hvis alle de nyeste passordene er unike skal den gratulere brukeren med en god jobb.

Eksempel der samme passord 'pwd1' brukes to ganger:

```
>>>> notebook = {'Facebook': ['Urgle', ['oldpwd', 'pwd1']], \
                 'reddit': ['Urgle', ['pwd1']]}
>>> secureSites(notebook)
You have used the password 'pwd1' at the following sites: Facebook, reddit.
```

Hvis du ikke har noen like passord:

```
>>> notebook = {'Facebook': ['Urgle', ['oldpwd', 'pwd1']], \
                 'reddit': ['Urgle', ['pwd2']]}
>>> secureSites(notebook)
No sites had similar passwords. Good job!
```

Skriv ditt svar her...

Maks poeng: 10

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 Error:

A try clause lets you execute code between the try: and except keywords. If an exception occurs, the code following the except keyword is executed. One important Error type is IOError, which is raised if an I/O operation (such as opening a file on disk) fails.

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 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 in k steps. 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).

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 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.close()

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

i **Kopi av Fotball**

Din tante har en særegen interesse for spesielle fotballkamper. Som familiens datakyndige har du blitt engasjert for å hjelpe henne med å automatisere hobbyen hennes. Hun har skrevet ned en drøss kamper i en tekstfil, på dette formatet: (som du også kan se i filen [Matches.txt](#))

```
Steinkjer, Byåsen, 3-5  
Byåsen, Steinkjer, 2-1  
Byåsen, Kvik Halden, 2-10  
Byåsen, Borg, 0-0  
...
```

Din jobb er å lage funksjoner som kan hjelpe henne med å få hente ut og tolke dette materialet.

12 Kopi av a) Lesing fra fil: (5%)

Lag funksjonen `importResults(file)`.

Alle resultater i serien ligger i en tekstfil kalt [Matches.txt](#) i den samme folderen som programmet, din funksjon skal lese fra denne filen. Hvis filen ikke eksisterer må du be brukeren om å skrive inn rett filnavn eller trykke 'q' for å avslutte. Innparameter til funksjonen skal være filnavn, og funksjonen skal returnere en liste der hver tekstlinje (med unntak av linjeskift) blir ett element i listen, se eksempelet under:

Kall og resultat:

```
>>> results = importResults('Matches.txt')
>>> print(results)
['Steinkjer,Byåsen,3-5', 'Byåsen,Steinkjer,2-1', 'Byåsen,Kvik Halden,2-10',
'Byåsen,Borg,0-0', 'Borg,Lade,3-2', 'Byåsen,Borg,0-0', 'Lade,Steinkjer,5-1',
'Nardo,Borg,3-3', 'Borg,Nardo,11-3', 'Steinkjer,Borg,2-2']
>>>
```

Hvis filnavn ikke finnes:

```
>>> results = importResults('Matches')
>>> 'Matches' could not be found. File name ('q' exits):
... og så avslutter funksjonen hvis filnavn er 'q', ellers går den videre som over og sjekker om rett filnavn igjen osv.
```

Skriv ditt svar her...

13 Kopi av b) Uthenting av resultater: (6%)

Lag en funksjon *analyzeResults(results)*

Funksjonen tar *results* fra oppgave a som input, og returnerer en *liste av lister* med 'hjemmelag', 'bortelag', hjemmemål og bortemål.

Eksempel på kall av funksjon og resultat:

```
>>> results = importResults('Matches.txt')
>>> analyzed = analyzeResults(results)
>>> print(analyzed)
[['Steinkjer', 'Byåsen', 3, 5], ['Byåsen', 'Steinkjer', 2, 1], ['Byåsen',
'Kvik Halden', 2, 10], ['Byåsen', 'Borg', 0, 0], ['Borg', 'Lade', 3, 2],
['Byåsen', 'Borg', 0, 0], ['Lade', 'Steinkjer', 5, 1], ['Nardo', 'Borg', 3,
3], ['Borg', 'Nardo', 11, 3], ['Steinkjer', 'Borg', 2, 2]]
```

Skriv ditt svar her...

1	
---	--

Maks poeng: 10

14 Kopi av c) Analyse av resultater: (3%)

Lag funksjonen `calculateScores(homeGoals, awayGoals)`.

Nå har du tilgang til alle kampene, og hvor mange mål hvert lag fikk. Din neste oppgave er å lage en funksjon som beregner poengene til de to ulike lagene i en enkelt kamp: Den skal ha som inputparametre målene fra en fotballkamp: (hjemmemål, bortemål). Funksjonen skal returnere to verdier: poengene til hjemmelaget og poengene til bortelaget. Tap gir 0 poeng, uavgjort gir 1 poeng, seier gir 3 poeng.

Eksempel for første kamp, der Byåsen slo Steinkjer på bortebane:

```
>>> match_result = calculateScores(3, 5)
>>> print(match_result)
(0, 3)
```

Skriv ditt svar her...

1	
---	--

Maks poeng: 10

15 Kopi av d) Analyse av resultat: (5%)

Lag funksjonen `sumTeamValues(analyzed)`.

Det neste du skal gjøre er å få oversikt over hvor mange poeng hvert enkelt lag har, og hvor mange kamper de har spilt. Funksjonen skal ha som input *analyzed* fra oppgave b. I retur skal du få en dictionary der nøklene er lagnavn som peker til en liste med lagets totale poengsum og totalt antall kamper. Det er nok lurt å bruke `calculateScores` fra forrige oppgave for å forenkle prosessen.

```
>>> analyzed = analyzeResults(results)
>>> team_data = sumTeamValues(analyzed)
>>> print(team_data)
{'Steinkjer': [1, 4], 'Byåsen': [8, 5], 'Kvik Halden': [3, 1], 'Borg': [10, 6], 'Lade': [3, 2], 'Nardo': [1, 2]}
```

Skriv ditt svar her...

1

Maks poeng: 10

16 Kopi av e) Sammenstilling av resultater: (5%)

Lag funksjonen `showResults(analyzed)`.

Din tante vil gjerne ha sammenstilt data på fine og lesbare måter. Hun vil ha sammenstilt alle kampdata på formatet du ser under. Hjemmeseier markeres med (H), uavgjort markeres med (U), mens borteseier markeres med (B). Funksjonen bruker de analyserte kampene fra oppgave b som input. Du kan forutsette at ingen lag har navn som er over fjorten tegn lange.

Funksjonen skriver ut tabellen under. Den er 45 tegn bred, feltene for hvert lagnavn skal være femten tegn lange. Legg også merke til hvordan målene er høyrestilt.

Eksempel på kjøring:

```
>>> showResults(analyzed)
#####
# Steinkjer      Byåsen      3 - 5 (B) #
# Byåsen        Steinkjer    2 - 1 (H) #
# Byåsen        Kvik Halden  2 - 10 (B) #
# Byåsen        Borg         0 - 0 (U) #
# Borg          Lade         3 - 2 (H) #
# Byåsen        Borg         0 - 0 (U) #
# Lade          Steinkjer    5 - 1 (H) #
# Nardo         Borg         3 - 3 (U) #
# Borg          Nardo        11 - 3 (H) #
# Steinkjer     Borg         2 - 2 (U) #
#####
```

Skriv ditt svar her...

Maks poeng: 10

17 Kopi av f) Sammenstilling av resultater: (6%)

Lag funksjonen `savePoints(team_data)`.

Nå skal du bruke resultatene fra `sumTeamValues` (dictionary i oppgave d) til å skrive ut en ordnet liste av alle lagene som har spilt, hvor mange poeng de har tilsammen og hvor mange kamper de har spilt. Dette skal lagres i filen `Points.txt` i den samme katalogen. Dersom filen finnes fra før skal du overskrive den. Filen skal se ut som vist under og i vedlegget [Points.txt](#). Det vil ikke legges vekt på helt perfekt treff på lengde av strenger, det er å vise at en forstår prinsippene som er viktig.

Eksempel på kjøring og resultat:

```
>>> savePoints(team_data)
Team information saved to Points.txt.
```

`Points.txt` vil da inneholde:

```
#####
# Navn          Poeng   Kamper  #
# Borg          10      6         #
# Byåsen        8       5         #
# Lade           3      2         #
# Kvik Halden   3       1         #
# Steinkjer     1       4         #
# Nardo         1       2         #
#####
```

Skriv ditt svar her...

Maks poeng: 10