NTNU
Det skapende universitet

Institutt for datateknikk og informasjonsvitenskap

Eksamensoppgave i TDT4110 Informasjonsteknologi – grunnkurs, kontinuasjonseksamen

Faglig kontakt under eksamen:

Tlf.:

Eksamensdato:

Eksamenstid (fra-til): **09:00 – 13:00**

Hjelpemiddelkode/Tillatte hjelpemidler: **Godkjent kalkulator**

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 lage løsning. Disponer tiden godt! Gjør rimelige antagelser der du mener oppgaveteksten er ufullstendig, skriv kort hva du antar.

Svar kort og klart, og skriv tydelig. Er svaret uklart eller lenger enn nødvendig trekker dette ned.

Målform/språk: **Bokmål**

Antall sider:

Innhold:

- Oppgave 1: Flervalgsoppgave (30 %)
- Oppgave 2: Grunnleggende programmering (10 %)
- Oppgave 3: Kodeforståelse (10 %)
- Oppgave 4: Mer programmering (50 %)
- Appendiks: Nyttige funksjoner
- Svarark til Flervalgsoppgave

Kontrollert av:

_____

Dato Sign

Oppgave 1: Flervalgsoppgave (30 %)

Bruk de to vedlagte svarskjemaene for å svare på denne oppgaven (ta vare på den ene selv). Du kan få nytt ark av eksamensvaktene dersom du trenger dette. Kun ett svar er helt riktig. 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.

1) **Hva er pseudokode?**
   - **a.** Foreløpig kode som ikke skal brukes i den endelige versjonen av et system
   - **b.** En diagramteknikk for å beskrive algoritmer eller programmer
   - **c.** De delene av et program som kan være årsak til en feil som er avdekket under testing
   - **d.** En måte å beskrive algoritmer som er mindre formell enn regulære programmeringsspråk
2) **Vi har 43 ulike symboler som vi ønsker å kode digitalt. Hva er det minste antall bits som vi må bruke?**
   - **a.** 5
   - **b.** 6
   - **c.** 7
   - **d.** 8
3) **Hvilken av disse aksessteknologiene gir høyest kapasitet ?**
   - **a.** Edge
   - **b.** 3G/UMTS
   - **c.** HSDPA
   - **d.** LTE
4) **Retningslinjene for algoritmer sier at det lønner seg å skrive en løkke for å utføre en repeterende oppgave hvis oppgaven må utføres**
   - **a.** mer enn 1 gang
   - **b.** 3-5 ganger eller mer
   - **c.** 8-10 ganger eller mer
   - **d.** 20 ganger eller mer
5) **Vi skal bruke binærsøking i en datamengde med 1 000 000 sorterte elementer. Hvor mange oppslag må vi vanligvis gjøre?**
   - **a.** 10
   - **b.** 20
   - **c.** 30
   - **d.** 50
6) **Analog informasjon er**
   - **a.** Diskret
   - **b.** Kontinuerlig
   - **c.** Tilfeldig
   - **d.** Digital
7) **Det er fire grunnleggende aktiviteter som inngår i alle programvareutviklingsprosesser.**
   - **a.** Spesifikasjon, utvikling, validering, evolusjon
   - **b.** Spesifikasjon, analyse, programmering, bruk
   - **c.** Forstudie, implementasjon, testing, bruk
   - **d.** Forstudie, programmering, simulering, vedlikehold

8) **Hvor mange negative og positive heltall kan vi representere i 2 byte?**

a. 2048
b. 32768
c. 65536
d. 1048576

9) Hvorfor kan et WiFi nettverk påvirke en Bluetooth forbindelse ?

a. WiFi nettverk har så sterkt signal
b. Bluetooth og WiFi bruker samme frekvensbånd
c. Bluetooth og WiFi bruker samme metode for koding
d. Ved samtidig bruk av WiFi og Bluetooth dannes et fenomen som heter "stående bølger"

10) Hvor mange instruksjoner er vanlig i en moderne datamaskin?

a. 3-7
b. 10-20
c. rundt 100
d. tusenvis

11) Hva kjennetegner problemløsning etter top-down prinsippet?

a. Vi løser de vanskeligste problemene først
b. Vi designer før vi programmerer en løsning
c. Vi går fra det generelle til det spesielle
d. Vi løser de enkelte delene av problemet først og bruker disse løsningene som byggeklosser i en generell løsning

12) Sortering ved innsetting (insertion sort) bruker omtrent 1 sekund på å sortere 1000 elementer. Hvor lang tid vil det omtrent ta å sortere 10000 elementer?

a. 2 s
b. 10 s
c. 100 s
d. 1000 s

13) Hva er pipelining?

a. Lignende operasjoner blir utført i rekkefølge
b. Flere instruksjoner kan være under utførelse samtidig
c. En effektiv organisering av datamaskinens hukommelse
d. En mekanisme for å utveksle data mellom programmer

14) Hvor mange symboler kan representeres av en sekvens på K bit?

a. K^2
b. 2^K
c. K
d. 2

15) Et nettverk sin evne til å levere uavbrutt / kontinuerlig tjeneste defineres som

a. Skalerbarhet
b. Tilgjengelighet
c. Pålitelighet
d. Ytelse

16) Ved digitalisering av lyd, hva oppnår vi med å bruke flere bit i hver punktprøve (sample)?

a. Vi trenger ikke å punktprøve (sample) like ofte
b. Vi oppfyller kravene i Nyquist-regelen
c. Vi kan representere stereo-lyd
d. Vi får en mer nøyaktig digitalisering

17) Det å utgi seg for være noen andre på internett kalles

a. Phishing
b. Spoofing
c. Pharming
d. Aliasing

18) Er det horisontale eller vertikale fargebånd som komprimeres bra i GIF-filer

a. Vertikale
b. Horisontale
c. GIF-filer komprimerer hverken vertikale eller horisontale fargebånd spesielt bra
d. GIF-filer komprimerer både horisontale og vertikale fargebånd veldig bra

19) Hente/Utføre-kretsløpet (Fetch/Execute Cycle) i en moderne datamaskin utføres

a. En gang i sekundet
b. Tusenvis av ganger i sekunder
c. Hundretusenvis av ganger i sekundet
d. Ikke oftere enn antall ganger klokken i maskina tikker

20) Hva kjennetegner inkrementell programvareutvikling?

a. Utvikler programvaren i de klart avskilte fasene krav, design, implementasjon, testing osv.
b. Utvikler programvaren uten noen plan eller prosess
c. Deler opp systemet i flere mindre deler som blir utviklet del for del
d. Lager en rask prototype basert på eksisterende krav

Oppgave 2: Grunnleggende programmering (10 %)

Oppgave 2a (5%)

Lag funksjonen `yatzy`. Den skal ha 5 innparametere, kalt t1, t2, t3, t4 og t5. Innparametrene representerer 5 tall mellom 1 og 6 (5 terninger).
Funksjonen skal returnere ei liste som inneholder de 5 tallene i sortert rekkefølge, eller en feilmelding hvis en av tallene er større enn 6 eller mindre enn 1.
Eksempel på kjøringer:

```
>>> yatzy(1,4,6,6,1)
[1, 1, 4, 6, 6]

>>> yatzy(1,4,6,4,10)
'Ikke bruk input stoerre enn 6!'

>>> yatzy(1,4,6,0,1)
'Ikke input verdier mindre enn 1!'
```

Oppgave 2b (5%)

Lag funksjonen `maxi_yatzy`. Den skal ta inn en liste med 5 eller 6 tall, og den skal returnere en skriftlig melding til brukeren som sier hvor mange terninger som ble kastet, hvilken verdi det var flest av, og hvor mange like det var av den verdien. Hvis det blir «uavgjort mellom to tall» brukes det høyeste tallet, slik som i dette eksempelet:

```
>>> maxi_yatzy([1,2,3,4,4,3])
'Du kastet 6 terninger og fikk flest 4 (2 like).'
```

Oppgave 3: Kodeforståelse (10 %)

Oppgave 3a (5%)

Gitt funksjonen `secret` vist under:

```
def secret(x):
    n = len(x)
    y = [0]*n

    for i in range(0,n):
        a = x[0]
        k = 0
        for j in range(1,n):
            if a < x[j]:
                a = x[j]
                k = j
        y[i] = a
        x[k] = -1
    return y
```

Hva returneres av etter å ha utført:

```
secret([1,4,8,2,5,8,10,1])
```

Oppgave 3b (5%)

Gitt funksjonen `secret2` vist under:

```
def secret2(x):

    a = 0

    while (x > 0):
        b = x%10
        a = a + 1
        x = (x - b) / 10

    y = a

    return y
```

Hva returneres av etter å ha utført:

```
secret2(12345678)
```

Oppgave 4: Mer programmering (50 %)

Oppgave 4a (5%)

Lag funksjonen *enterWords* som har ingen innparametre, og returnerer ei liste, *wordList*, med ord som brukeren selv har skrevet inn. Brukeren kan avslutte innskriving av ord ved å trykke enter uten å skrive noe. Prompten til brukeren skal vise "Enter word [press Enter to quit]:"

Eksempel på kjøring (ord med fet skrift er ord brukeren selv har skrevet inn):

```
>>> liste=enterWords()
Enter word [Press Enter to quit]: house
Enter word [Press Enter to quit]: chair
Enter word [Press Enter to quit]: university
Enter word [Press Enter to quit]: mouse
Enter word [Press Enter to quit]:
>>> liste
['house', 'chair', 'university', 'mouse']
```

Oppgave 4b (10%)

Lag funksjonen *noVowels*, som tar inn ei liste med ord (*inList*), og returnerer en ny liste, *outList*, med ord der alle vokalene i det engelske alfabetet (bokstavene "a", "e", "i","o","u" og "y") fra *inList* er fjernet.

Eksempel på kjøring (liste har samme verdi som i oppgave 4a):

```
>>> nyliste=noVowels(liste)
>>> nyliste
['hs', 'chr', 'nvrst', 'ms']
```

Oppgave 4c (10%)

Lag funksjonen, *randomSequence*, som tar inn to lister (*listOne*, *listTwo*) og returnerer to lister (*newlistOne*, *newlistTwo*) der rekkefølgen på elementene i begge listene tilfeldig endret på samme måte. Det vil si at f.eks. element 3 i både *listOne* og *listTwo* for blir plassert som element 1 i både *newlistOne* og *newlistTwo*. Du har ikke lov til å bruke funksjoner som *random.shuffle()* til å stokke om på rekkefølgen i lister.

Eksempel på kjøring (*liste* og *nyliste* har samme verdier som i oppgave 4a/4b):

```
>>> (svar,puzzle) = randomSequence(liste,nyliste)
>>> svar
['university', 'house', 'chair', 'mouse']
>>> puzzle
['nvrst', 'hs', 'chr', 'ms']
>>>
```

Oppgave 4d (5%)
Lag funksjonen *printNewlines* som tar inn en variabel, *number*, og skriver ut *number* antall linjeskift til konsoll.

Eksempel på kjøring:
```
>>> printNewlines(5)




>>>
```

Oppgave 4e (10%)
Lag funksjonen playGame som tar inn to lister (*answers*, *puzzles*), der brukeren skal bli presentert for ett og ett ord i lista *puzzles* (ord uten vokaler), gjette hvilket ord det er som sjekkes mot ord i lista *answers*. Riktige svar belønnes med 1 poeng. Funksjonen skal returnere antall poeng brukeren scorer i variabelen *points*. Hva som skal skrives ut til skjerm er vist i eksempelet på kjøring under. Ord i eksemplet skrevet med fet skrift er ord skrevet inn av brukeren.

Eksempel på kjøring (verdier fra listene er hentet for tidligere oppgaver):
```
>>> poeng = playGame(svar,puzzle)
Puzzle word: nvrst
Guess word? university
You answered correctly!
Puzzle word: hs
Guess word? hose
You answered incorrectly! The answer should be house
Puzzle word: chr
Guess word? chair
You answered correctly!
Puzzle word: ms
Guess word? mouse
You answered correctly!
>>> poeng
3
```

Oppgave 4f (10%)
Skrive koden for å gjøre ferdig spillet *NoVowels* game. Spillet skal gjøre følgende:
1) Skrive ut følgende til skjerm:

```
The NoVowels Game
=================================
Player 2: Look away from the screen
Player 1: Write in a list of English words in lower-case.
```

2) Hente ei liste med ord som brukeren selv skriver inn og lagre dette i variabelen `wordList`.

3) Lage en ny liste, `noVowelsList`, som tilsvarer lista `wordList` uten vokaler.

4) Lage to nye lister, `answers` og `quizzes`, der rekkefølgen på ordene i listene `wordList` og `noVowelsList` er byttet om vilkårlig, men på samme vis for begge listene.

5) Skriv ut 50 linjeskift til konsoll

6) Skriv ut til skjerm følgende:

```
Player 2: Guess words that lack all vowels:
```

7) Spill igjennom gjetting av ord med listene `answers` og `quizzes` og lagre poengsummen i variabelen `points`.

8) Skriv ut til skjerm hvor mange poeng spilleren fikk av antall mulige:

```
You've got 3 of 4 points
```

Utskrift av komplett kjøring (tekst fra bruker med fet skrift):

```
The NoVowels Game
=================================
Player 2: Look away from the screen
Player 1: Write in a list of English words in lower-case.
Enter word [Press Enter to quit]: house
Enter word [Press Enter to quit]: chair
Enter word [Press Enter to quit]: university
Enter word [Press Enter to quit]: mouse
Enter word [Press Enter to quit]:
```

(50 linjeskift her)

```
Player 2: Guess words that lack all vowels:
Puzzle word: chr
Guess word? chair
You answered correctly!
Puzzle word: hs
Guess word? host
You answered incorrectly! The answer should be house
Puzzle word: ms
Guess word? mouse
You answered correctly!
Puzzle word: nvrst
Guess word? university
You answered correctly!
You've got 3 of 4 points
```

Appendiks: Nyttige funksjoner/metoder i Python

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.

%
: Reminder: 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.

str([object])
: Return a string containing a nicely printable representation of an object.

pow(x, y)
: Return x to the power y (x**y or x^y)

Library: math

math.pow(x,y)
: Return x to the power y (x**y or x^y)

math.sqrt(x)
: Return the square root of x.

math.pi
: The mathematical constant π = 3.141592...

math.e
: The mathematical constant e = 2.718281...

Library: random

random.randint(a, b)
: Return a random integer N such that a <= N <= b.

random.random()
: Return the next random floating point number in the range 0 <= N < 1.

String methods:

s.isalnum()
: Returns true if the string contains only alphabetic letters or digits and is ate 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.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.

List operations:

s[i:j]
: Return slice starting at position i extending to position j. 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:

clear()
: Clears the contents of a dictionary

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.

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

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

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.1

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

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

Svarskjema flervalgsoppgave

Kandidatnummer: _____ Program: _____

Fagkode: _____ Dato: _____

Antall sider: _____ Side: _____

Oppgavenr	***A***	***B***	***C***	***D***
1.1				
1.2				
1.3				
1.4				
1.5				
1.6				
1.7				
1.8				
1.9				
1.10				
1.11				
1.12				
1.13				
1.14				
1.15				
1.16				
1.17				
1.18				
1.19				
1.20				

Svarskjema flervalgsoppgave

Kandidatnummer: _____ Program: _____

Fagkode: _____ Dato: _____

Antall sider: _____ Side: _____

Oppgavenr	***A***	***B***	***C***	***D***
1.1				
1.2				
1.3				
1.4				
1.5				
1.6				
1.7				
1.8				
1.9				
1.10				
1.11				
1.12				
1.13				
1.14				
1.15				
1.16				
1.17				
1.18				
1.19				
1.20				

Løsningsforslag

Oppgave 1 (30%)

Oppgavenr	*A*	*B*	*C*	*D*
1.1				
1.2				
1.3				
1.4				
1.5				
1.6				
1.7				
1.8				
1.9				
1.10				
1.11				
1.12				
1.13				
1.14				
1.15				
1.16				
1.17				
1.18				
1.19				
1.20				

Oppgave 2 (10%)

Oppgave 2a (5%)

```
def yatzy(t1, t2, t3, t4, t5):
    liste = [t1,t2,t3,t4,t5]
    liste.sort()
    if (max(liste)>6):
        liste = "Ikke bruk input stoerre enn 6!"
    elif (min(liste)<1):
        liste = "Ikke bruk input mindre enn 1!"
    return  liste
```

Oppgave 2b (5%)

```
def maxi_yatzy(liste):
    flestverdi = 0
    maxantall = 0

    for i in range(1,7):
        antall = 0
        for item in liste:
            if (item==i):
                antall = antall + 1
        if (antall >= maxantall):
            flestverdi = i
            maxantall = antall
    melding = "Du kastet "+str(len(liste))+" terninger og fikk flest "
    melding = melding + str(flestverdi)+" ("+str(maxantall)+" like)."
    return melding
```

Oppgave 3 (10%)

Oppgave 3a (5%)

```
>>> secret([1,4,8,2,5,8,10,1])
[10, 8, 8, 5, 4, 2, 1, 1]
```

Oppgave 3b (5%)

```
>>> secret2(12345678)
8
```

Oppgave 4 (50%)

Oppgave 4a (5%)

```
def enterWords():
    wordList=[]
    result = "something" # To avoid the while-loop to be true first time
    while (result !=""):
        result = input("Enter word [Press Enter to quit]: ")
        if (result != ""):
            wordList.append(result)
    return wordList
```

Oppgave 4b (10%)

```
def noVowels(inList):
    vowels=["a","e","i","o","u","y"]
    outList = []
    for word in inList:
        newWord = ""
        for letter in word:
            if (letter not in vowels):
                newWord = newWord + letter
        outList.append(newWord)
    return outList
```

Oppgave 4c (10%)

```
import random

def randomSequence(listOne,listTwo):
    newlistOne = []
    newlistTwo = []
    while(len(listOne)):
        number = random.randint(0,len(listOne)-1)
        newlistOne.append(listOne[number])
        newlistTwo.append(listTwo[number])
        del listOne[number]
        del listTwo[number]
    return [newlistOne,newlistTwo]
```

Oppgave 4d (5%)

```
def printNewlines(number):
    for i in range(1,number):
        print()
```

Oppgave 4e (10%)

```
def playGame(answers, puzzles):
    points = 0
    for i in range(0,len(answers)):
        print("Puzzle word:",puzzles[i])
        answer = input("Guess word? ")
        if (answer == answers[i]):
            print("You answered correctly!")
            points = points + 1
        else:
            print("You answered incorrectly! The answer should be",answers[i])
    return points
```

Oppgave 4f (10%)

```
# Print information to the screen
print("The NoVowels Game")
print("==================================")
print("Player 2: Look away from the screen")
print("Player 1: Write in a list of English words in lower-case.")

# Get a list of words entered by Player 1
wordList = enterWords()

# Create a new list without vowels
noVowelsList = noVowels(wordList)

# Randomize the sequence of words for both lists
[answers,quizzes] = randomSequence(wordList,noVowelsList)

# Print 50 newlines
printNewlines(50)

# Play the game
print("Player 2: Guess words that lack all vowels:")
points = playGame(answers,quizzes)
print("You've got",points,"of",len(answers),"points")
```