MATEMATISKA INSTITUTIONEN
STOCKHOLMS UNIVERSITET
Avd. Beräkningsmatematik
Examinator: Lars Arvestad

Tentamensskrivning i
DA2005 Programmeringsteknik
och DA2004 Programmeringsteknik för matematiker
7.5 hp
2022-08-22

- Inga hjälpmedel tillåtna.
- **Skriv tydligt.** Svårlästa svar riskerar 0 poäng.
- Skriv bara på en sida av varje papper!
- Motivera alla svar (om inte annat anges)!
- Man måste bli godkänd på del A (5 rätt på 10 frågor) för att del B ska rättas.
- **Hjälpmedel:** Ett A4 med så mycket information du vill. Du får skriva på båda sidorna.
- **Betygsgränser:** E: 10, D: 12, C: 14, B: 16, A: 18, av maximala 20.

Del A: flervalsfrågor

Var snäll samla svaren på del A på ett svarspapper.

1. Vad menas med "typ" när man programmerar?

A. Det avser elementen i en lista: "Tag typ 1 från listan."
B. Det är en beskrivning av ett värde. "Parametern ska rätt typ."
C. Det är en dålig programmerare. "Den här koden skrevs av en typ."
D. Det är ett sorts fel. "En typ uppstod, så programmet avbryts."
E. Det är en funktion som kan skrivas på en rad. "Skriv en typ som returnerar..."

2. Vilka operatorer kan användas på boolska värden? Mer än ett svar är korrekt.

A. `and`
B. `or`
C. `not`
D. `almost`
E. `probably`

3. Betrakta kodsnutten till höger. Vad skrivs ut?

A. 0
B. 2
C. 7
D. 8
E. 56

```
x = 1
y = 0

def f(x):
    x = x + 1
    return x

def g(y):
    return x * y

x = g(f(7))
print(x)
```

4. Betrakta kodsnutten till höger. Vad blir resultatet av anropet `f([1,2,3,4])`?

A. `[1,2,3,4]`

B. `[4,3,2,1]`

C. `[2, 3]`

D. `[1]`

E. `[4]`

```
def f(lst):
    if lst == []:
        return []
    else:
        return [lst[-1]] + f(lst[:-1])
```

5. Betrakta kodsnutten till höger. Vad blir resultatet av anropet `b('hubba', 'abc')`?

A. `'hu***'`

B. `'hu'`

C. `['h', 'u', '*', '*', '*']`

D. `[True, True, False, False, False]`

E. Inget, det uppstår ett fel på grund av att funktionen `check` är definierad på fel plats.

```
def b(s, t):
    def check(x):
        if x in t:
            return '*'
        else:
            return x

    return [check(x) for x in s]
```

6. Betrakta kodsnutten till höger. Vad blir resultatet av anropet `g('')`?

A. `0`

B. `10`

C. `'x'`

D. `'xxxxxxxxxx'`

E. Inget, det uppstår ett fel.

```
a = 10
def g():
    res = ''
    while a > 0:
        res += 'x'
        g(res)
    return res
```

7. Vilken av följande kodrader är ett lämpligt sätt att signalera att ett fel har uppstått?

A. **raise** `Exception('Fel uppstod')`

B. **assert** `Exception('Fel uppstod')`

C. **return** `'Fel uppstod'`

D. **return** `Exception('Fel uppstod')`

E. **return** `None`

8. Vilket begrepp brukar användas för funktioner som tar funktioner som parametrar?

A. Högre ordningens funktioner.

B. Objektorientering.

C. Metoder.

D. Iteratorer.

E. Konstruktorer.

9. Betrakta kodsnutten till höger. Vad blir resultatet av sista raden?

A. `7`

B. `17`

C. Inget, det uppstår ett körfel på grund av att man inte kan instantiera med `b = B()` här.

D. Inget, det uppstår ett körfel (*runtime error*) på grund av att `m` inte är definierat för B.

E. Inget, det uppstår ett syntaxfel.

```
class A:
    def m(self):
        return self.num

class B(A):
    def n(self):
        print(self.num)

a = A()
b = B()
a.num = 7
b.num = 17
print(b.m())
```

10. Vilket av följande påståenden är *inte* en bra tumregel när man programmerar?

A. Försök skriva korta funktioner!

B. Begränsa antalet funktioner!

C. Använd beskrivande identifierare!

D. Använd inte globala variabler i onödan!

E. Funktioner bör returnera värden!

Del B: kodfrågor

Var snäll använd ett papper (eller fler) till varje fråga i del B.

11. I den här uppgiften ska vi jobba med enkel textanalys.

A. I figur 1 finns ett försök till en funktion som tar bort "stoppord" från en text. Stoppord är ord som betraktas som ointressanta för en textanalys eftersom de inte bidrar med information. Vanliga prepositioner är, till exempel, ofta inte meningsfulla att ta med i en textanalys. Anropet `remove_stop_words (text, stopwords)` ska filtrera bord stoppord. Om `stop_words = {'en': True, 'ett': True, 'i': True, 'på': True}` så ska funktionen fungera så här:

```
[In: ] remove_stop_words(['en', 'enkel', 'text', 'på', 'svenska'], stop_words)
[Out:] ['enkel', 'text', 'svenska']
```

Tyvärr har funktionen minst två fel! Vilka? (2p)

```
def remove_stop_words(text, stopwords):
    '''
    Return a version of the text without the "forbidden" words defined by
    the second parameter.

    text: a list of words representing a text.
    stopwords: a dictionary containing words that we do not want.
    '''
    # Note: broken code!
    for word in text:
        if word in stopwords:
            result.append(word)
    return result
```

Figur 1: Problematisk kod för uppgift 11A. Vad är fel?

B. Skriv en funktion `term_frequency(text)` som beräknar $\mathrm{tf}(w)$, den relativa frekvensen för alla ord w i en text given som en lista. Om n_w är antalet gånger ordet w finns i en text och N är antalet ord i texten så är $\mathrm{tf}(w) = n_w/N$.

Låt `term_frequency(text)` returnera en uppslagstabell med ord som nycklar och deras $\mathrm{tf}(w)$ som associerade värden. Du behöver inte bekymra dig över ordformer.

Exempelanvändning:

```
[In: ] text = ['alla', 'kan', 'baka', 'alla', 'recept']
[In: ] term_frequency(text)
[Out:] {'alla': 0.4, 'kan': 0.2, 'baka': 0.2, 'recept': 0.2}
[In: ] term_frequency([])
[Out:] {}
```

(2p)

C. Skriv funktionen `top_word(tf, importance)` som returnerar det ord i `tf` som är viktigast, definierat som ordets termfrekvens multiplicerat med en vikt som man hittar i uppslagstabellen `importance`. Med andra ord, ordet w har "viktigheten" `tf[w] * importance[w]`. Om w inte finns i `importance` så ska vikten 1 användas.

Om det finns mer än ett ord med samma högsta viktighet så returneras ett av dem, godtyckligt valt.

Exempelanvändning:

```
[In: ] text = ['alla', 'kan', 'baka', 'alla', 'recept']
[In: ] word_imp = {'baka': 3, 'recept': 2}
[In: ] tf = term_frequency(text)
[In: ] top_word(tf, {})
[Out:] 'alla'
[In: ] top_word(tf, word_imp)
[Out:] 'baka'
```

(2p)

12. Denna sommar har man i lokaltrafiken kunnat se reklam från ett företag som vill rekrytera programmerare. De söker personer som kan lösa deras programmeringsuppgift.

Om man översätter företagets pseudokod till Python så kan resultatet bli som här nedanför. Vad skrivs ut av programmet? (2p)

```
def fun(n):
    if n == 0:
        return 2
    elif n == 1:
        return 1
    else:
        return fun(n - 1) + fun(n - 2)

s = ''
for i in range(6):
    s += str(fun(i))
print(s)
```

13. Skriv funktionen `filter2(predicate, iterable)` som returnerar två listor: den första listan innehåller de element från `iterable` som funktionen `predicate` returnerar `True` (eller motsvarande) för, och den andra listan de övriga elementen.

Det är alltså en variant på den inbyggda funktionen **`filter`**, som bara returnerar en lista (med de element som predikatet är sant för).

Krav:

- Du får inte använda **`filter`** som hjälpfunktion.
- Funktionen ska fungera som i exemplen nedan.

Om man ska vara noggrann så returnerar **`filter`** en iterator, men vi förenklar och returnerar en lista.
Exempelanvändning:

```
[In: ] filter2(str, [])
[Out:] ([], [])
[In: ] filter2(lambda x: x >= 0, [-2, -1, 0, 1, 2])
[Out:] ([0, 1, 2], [-2, -1])
[In: ] filter2(lambda x: x >= 0, [-2, -1])
[Out:] ([], [-2, -1])
[In: ] def even(x): return x % 2 == 0
[In: ] filter2(even, [0, 1, 2, 3, 4])
[Out:] ([0, 2, 4], [1, 3])
```

(2p)