MATEMATISKA INSTITUTIONEN
STOCKHOLMS UNIVERSITET
Avd. Beräkningsmatematik
Kursledare: Anders Mörtberg (DA2004) och
Lars Arvestad (DA2005)
Examinator: Lars Arvestad

Tentamensskrivning i
DA2004 Programmeringsteknik för matematiker *och*
DA2005 Programmeringsteknik
7.5 hp
2024-02-12

- Tentan har flervalsfrågor där minst ett svarsalternativ är korrekt. Om man svarar fel eller inte har exakt antal rätta alternativ får man noll poäng på frågan.
- **Skriv tydligt.** Svårlästa svar riskerar 0 poäng.
- Skriv bara på en sida av varje papper!
- Man måste bli godkänd på del A (4 rätt på 8 frågor) för att del B ska rättas.
- Del B består av ett antal frågor med varierande poäng (totalt 12).
- Inga `import` (Pythons standardbibliotek eller externa bibliotek) får användas om de inte nämns eller finns med i uppgiften. Man får dock använda inbyggda funktioner som `len`, `range` och `map`.
- All kod avser **Python 3**, dvs *inte* t.ex. Python 2.7
- **Hjälpmedel:** Ett hand- eller datorskrivet A4 med så mycket information du vill på både fram och baksida.
- **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 är resultatet av `float(int(float('3.14')))`?

A. Heltalet `0`
B. Heltalet `3`
C. Flyttalet `0.0`
D. Flyttalet `3.0`
E. Flyttalet `3.14`

2. Om `d = { 3 : [4, 'e', 2], 'omtenta' : { 'e' : 2, 3 : 'ja' } }`, vilket eller vilka av följande alternativ kan man skriva för att `2` ska skrivas ut?

A. `print(len(d))`
B. `print(d[3][2])`
C. `print(d['omtenta']['e'])`
D. `print(len(d['omtenta'][3]))`
E. `print(len(d['omtenta']))`

3. Vad händer om man kör koden till höger och användaren skriver in `42`?

A. `Even` skrivs ut och loopen avslutas.
B. `Even` skrivs ut och loopen börjar om.
C. `Odd` skrivs ut och loopen avslutas.
D. `Odd` skrivs ut och loopen börjar om.
E. Inget av alternativen A.-D. då ett särfall lyfts.

```
while True:

    x = int(input("Write a number: "))

    if x % 2 == 0:
        print("Even")
    else:
        print ("Odd")
```

4. Vad gör `raise` i Python?

A. Det används för att kasta ett särfall.
B. Det används för att upphöja ett tal med ett annat tal.
C. Det används för att generera slumpmässiga tal.
D. Det används för att skapa en generator.
E. Det finns inget som heter `raise` i Python.

5. Vad är `sys` i Python?

A. Det är en funktion för att läsa av tangentbordet utan att vänta på returtangenten.
B. Det är en trigonometrisk funktion.
C. Det är det kommando som startar en interaktiv session med Python, som Console i Spyder.
D. Det är en modul med bland annat funktioner för Python-systemet.
E. Det är ett felmeddelande.

6. Vad skrivs ut om vi kör koden nedan?

```
xs = [1, 2, 3]
ys = [4, 5]

print(len([ x + y for x in xs for y in ys ]))
```

A. `0`
B. `5`
C. `6`
D. `15`
E. Inget av alternativen A.-D. då ett särfall lyfts.

7. Vilken eller vilka boolska variabeltilldelningar gör att uttrycket `not x and (x or not y)` evaluerar till `True`?

A. `x = False, y = False`
B. `x = False, y = True`
C. `x = True, y = False`
D. `x = True, y = True`
E. Ingen, det evaluerar alltid till `False`.

8. Vad skrivs ut om vi kör koden till höger?

A. `-20`
B. `-10`
C. `10`
D. `30`
E. `50`

```
x = 30
y = 20

def f(x):
    if x >= y:
        return x + y
    else:
        return x - y

print(f(10))
```

Del B: kodfrågor

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

9. I koden nedan ges en funktion, `integrate(fcn, a, b, n)`, för att numeriskt bestämma värdet på integralen för en funktion `fcn` från `a` till `b` med hjälp av rektangelmetoden som du lärde dig om på gymnasiet (`n` anger hur många rektanglar som ska användas). Med anropet

```
integrate(lambda x: x**2, 0, 1, 100)
```

approximeras alltså $\int_0^1 x^2 dx$ genom att summera arean på 100 rektanglar under funktionen x^2, dvs

$$\sum_{i=0}^{n-1} d(a + i \cdot d)^2$$

där d är det steg som man tar: $d = (b - a)/n$. Summanden $d(a + i \cdot d)^2$ är alltså en rektangel med bas d och höjd $(a + i \cdot d)^2$ (där upphöjt i 2 kommer från att vi approximerar integralen av x^2).

Matematiskt är funktionen korrekt, men koden nedan har (minst) två fel. Vilka? **(2p)**

```
def integrate(fcn, a, b, n):
    d = (b - a) / n
    x = a
    for i in range(n):
        y_at_x = fcn(x)
        integral_approximation += y * d
        x += d
    return integral_approximation
```

10. Skriv funktionen `text2morse(s)` som översätter strängen `s` till morsekod. Funktionen kan ignorera andra tecken än bokstäver och ska använda uppslagstabellen `morse` i Figur 1.

Varje bokstav, översatt till morse, ska separeras med ett mellanslag så att man kan se var morsebokstäverna börjar och slutar. Ett mellanslag i indata ska översättas till tre mellanslag.

Du kan anta att indata ges som versaler (stora bokstäver) och att ord separeras med ett mellanslag. Du behöver inte skriva av koden i Figur 1 utan kan använda `morse` fritt i din kod. **(2p)**

Exempelanvändning, med mellanslag särskilt markerade som ␣:

```
[In: ] text2morse('SOS')
[Out:] '...␣---␣...'
[In: ] text2morse('INTE')
[Out:] '..␣-.␣-␣.'
[In: ] text2morse('HEJ')
[Out:] '....␣.␣.---'
[In: ] text2morse('HOPP')
[Out:] '....␣---␣.--.␣.--.'
[In: ] text2morse('HEJ␣HOPP')
[Out:] '....␣.␣.---␣␣␣....␣---␣.--.␣.--.'
```

```
morse = {'A': '.-',   'B': '-...',  'C': '-.-.', 'D': '-..',  'E': '.',
         'F': '..-.', 'G': '--.',   'H': '....', 'I': '..',   'J': '.---',
         'K': '-.-',  'L': '.-..',  'M': '--',   'N': '-.',   'O': '---',
         'P': '.--.', 'Q': '--.-',  'R': '.-.',  'S': '...',  'T': '-',
         'U': '..-',  'V': '...-',  'W': '.--',  'X': '-..-', 'Y': '-.--',
         'Z': '--..', 'Å': '.--.-', 'Ä': '.-.-', 'Ö': '---.'}
```

Figur 1: Morsealfabetet för svenska representerat i Python-kod. Vi bortser här från siffror, skiljetecken och annat. Versaler och gemener översätts på samma sätt i Morsealfabetet, men det räcker att du hanterar versaler i den här uppgiften.

11. Skriv en funktion `morse_translate(infilename, outfilename)` som öppnar filen `infilename`, läser in den text som kan finnas där över flera rader, och översätter till morsekod som skrivs ut på en fil som ges av `outfilename`.

Om det uppstår något fel, till exempel på grund av att infilen inte finns, så ska meddelandet `"Error (. .-. .-. --- .-.)"` skrivas ut.

Din lösning ska använda funktionen `text2morse` från fråga 10 (oavsett om du löst den uppgiften eller inte). Tänk på att raderna i filen avslutas med `\n` (nyrad) vilket troligtvis inte stöds av `text2morse`. **(2p)**

12. Skriv en högre ordningens funktion `intersection_with(list1,list2,f)` som given listorna `list1` och `list2`, samt en binär funktion `f` som returnerar en **`bool`**, returnerar alla par av `(x,y)` för vilka `f(x,y)` är `True`. **(2p)**

Exempelanvändning:

```
[In: ] print(intersection_with([1,4,2,3],[5,3,6,2],lambda x, y: x == y))
[Out:] [(2, 2), (3, 3)]
[In: ] print(intersection_with([1,4,2,3],[5,3,6,2],lambda x, y: x > y))
[Out:] [(4, 3), (4, 2), (3, 2)]
```

13. På ordinarie tenta 2024-01-04 skulle man skriva en klass för att representera rationella tal (bråktal). Nedan följer en exempellösning:

```
from math import gcd

class Rational:

    def __init__(self,n,d):
        if d == 0:
            raise ValueError("Cannot have denominator 0")
        else:
            self.numerator = n
            self.denominator = d

    def __str__(self):
        return str(self.numerator) + " / " + str(self.denominator)

    def normalize(self):
        g = gcd(self.numerator,self.denominator)
        self.numerator //= g
        self.denominator //= g
```

A. Lägg till metoden `mul` för att multiplicera ett rationellt tal med ett annat rationellt tal. **(2p)**

B. Lägg till metoden `add` för att addera ett rationellt tal med ett annat rationellt tal. **(2p)**

Obs: ni behöver inte normalisera talen efter addition/multiplikation, men additionen ska så klart fungera på korrekt sätt för tal med olika nämnare. Metoderna ska ha som sidoeffekt att talet ändras när man adderar/multiplicerar med ett tal (se exemplen nedan).

Exempelanvändning:

```
[In: ] r = Rational(2,4)       # r := 2 / 4
[In: ] r.mul(r)                # r := r * r
[In: ] print(r)
[Out:] 4 / 16
[In: ] r.normalize()
[In: ] print(r)
[Out:] 1 / 4
[In: ] r.add(r)                # r := r + r
[In: ] print(r)
[Out:] 8 / 16
[In: ] s = Rational(2,3)       # s := 2 / 3
[In: ] s.add(r)                # s := s + r
[In: ] print(s)
[Out:] 56 / 48
```