MATEMATISKA INSTITUTIONEN
STOCKHOLMS UNIVERSITET
Avd. Beräkningsmatematik
Kursledare: Lars Arvestad
Examinator: Lars Arvestad

Tentamensskrivning i
DA2005 Programmeringsteknik och **DA2004** … för matematiker
7.5 hp
2021-08-23

- Del A har flervalsfrågor där minst ett svarsalternativ är korrekt. Om man svarar fel eller inte har exakt rätt antal alternativ får man 0 poäng på frågan.
- Man måste bli godkänd på del A (4 poäng på 8 frågor) för att få göra del B.
- Del B består av ett antal frågor med varierande poäng (totalt 12) vilka ska lösas genom att man skriver Python-kod.
- Svaren till del B lämnas in i en `.py`-fil namngiven `anonymkod.py` där `anonymkod` är koden som man får i Ladok när man registrerar sig till tentan. Man **måste** även ha med sin anonyma kod i en kommentar i toppen av filen. Man ska **inte** skriva sitt riktiga namn någonstans i filen!

  Skriv också gärna anonymiseringskod i kommentar högst upp i filen.
- Inlämning sker på kurswebben för den senaste kursomgång (DA2004 *eller* DA2005) som du har gått.

  Observera! Om det är problem med inlämningssystemet så är *reservförfarandet* att man mejlar in sina lösningar till studentexpeditionen. Men: du måste då placera din Python-fil i en zip-fil eftersom universitetets mejlsystem blockerar Python-filer.
- Var *noga* med att **namnge** *funktioner och klasser* rätt (på samma sätt som de är namngivna i uppgiften).
- Inga `import` 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:** Till del A får man ha ett A4 med så mycket information man vill. Du får skriva på båda sidorna. Del B är öppen bok då det är hemtenta och samma regler kring hjälpmedel gäller som för projekt och labbar.
- **Betygsgränser:** E: 10, D: 12, C: 14, B: 16, A: 18, av maximala 20.

Del A: flervalsfrågor (1p per fråga)

1. Vilka typer är funktionen `len` är tillämpbar på?

A. str
B. list
C. dict
D. tupel (som t.ex. `(1, 'a', False)`)
E. int

2. Vad skrivs ut av koden till höger?

A. 0
B. 1
C. 2
D. 3
E. 4

```
a = 1
b = 2
def f(x):
    a = 3
    b = x + a
    return b
def g(a, b):
    x = f(a + b)
    return x

print(g(0, 1))
```

3. Vad ger `[x ** 2 for x in range(5)]` för resultat?

A. `[0, 1, 2, 3, 4]`

B. `[1, 2, 3, 4, 5]`

C. `[0, 2, 4, 6, 8]`

D. `[2, 4, 6, 8, 10]`

E. `[0, 1, 4, 9, 16]`

4. Vad används `lambda` till?

A. Instruktionen används för att definiera funktioner.
B. Beräkna hur mycket minne som används av en datastruktur.
C. Instruktionen används för felsökning.
D. Instruktionen används för att analysera snabbhet.
E. Det är en datastruktur för balanserade träd.

5. Hur många kombinationer av tilldelningar av `True` / `False` för variablerna `x`, `y`, och `z` finns det som gör uttrycket `x and (y or z)and (not y or not z)` sant?

A. 0
B. 1
C. 2
D. 3
E. 4

6. Hur stora heltal kan man representera i beräkningar i Python?

A. $[-2^{16}, 2^{16} - 1]$
B. $[-2^{32}, 2^{32} - 1]$
C. $[-2^{64}, 2^{64} - 1]$
D. $[-2^{128}, 2^{128} - 1]$
E. Ingen begränsning, förutom datorns minne.

7. Om funktionen `fkn` är definierad som till höger, vad returnerar anropet `fkn(4)` ?

```
def fkn(n):
    a = 0
    b = 1
    for i in range(n):
        c = a + b
        a = b
        b = c
    return b
```

A. 2
B. 3
C. 4
D. 5
E. 6

8. Vad har instruktionen `raise` för syfte?

A. Utföra exponentiering av stora tal.
B. Skapa alternativa identifierare när man importerar moduler.
C. Skapa ett *exception* för att signalera problem.
D. Höja prioriteten på en beräkning, så att andra beräkningar pausar.
E. Det används i `print`-satser, för att till exempel kunna göra en fotnot (t.ex. ”hubba[1]”).

Del B: kodfrågor (2p per fråga)

9. Klassen `Point` till höger kan man använda för att representera punkter i ett kartesiskt koordinatsystem.

- Lägg till en metod `translate` som tar parametrarna `dx` och `dy` som ändrar punktens position från (x, y) till $(x + d_x, y + d_y)$.
- Visa hur man instantierar en punkt i $(0, 0)$ och translaterar den med $(d_x, d_y) = (1, 1)$.

```
class Point:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def __str__(self):
        return f'<Point ({x}, {y}>'
```

10. Skriv en subklass till `Point` (från fråga 9) som heter `PolarPoint` och har en konstruktor som använder polära koordinater, men konverterar internt till kartesiska koordinater.

Med hjälp av arv ska man kunna använda funktionen `translate`, definierade i klassen `Point`, även i `PolarPoint`.

Påminnelse: om de polära koordinaterna ges av radien r och vinkeln α, så är $x = r \cos \alpha$ och $y = r \sin \alpha$.

I den här uppgiften bör du använda modulen `math`.

11. Skriv en funktion `punctuation(filename)` som läser en text från en fil med namnet `filename` och returnerar en sträng där allt förutom skiljetecken (punkt, komma, kolon, semikolon, utropstecken, frågetecken) har tagits bort.

Om filen ”tonåring.txt” innehåller texten

> Öh, alltså farsan,
> det där suger!

så ska anropet `punctuation('tonåring.txt')` returnera strängen `',,!'`.

12. Skriv en funktion `swe_to_ascii` som tar en sträng s som indata och returnerar en liknande sträng, men där de svenska tecknen ”å”, ”ä”, och ”ö” har ersatts av ”aa”, ”ae”, och ”oe”. Du behöver bara hantera gemena bokstäver, versaler kan ignoreras.

Tester:

```
[In] : swe_to_ascii('')
[Out]: ''
[In] : swe_to_ascii('åäö')
[Out]: 'aaaeoe'
[In] : swe_to_ascii('Jag är mållös.')
[Out]: 'Jag aer maalloes.'
```

Not: ASCII är namnet på den gamla standard som ibland fortfarande används för att koda text, men inte har svenska tecken (förutom i en anpassad standard). Numer använder vi framförallt UTF-8 där många nationella bokstäver återfinns.

13. Vi vill packa föremål i lådor. Föremålen som ska packas följer ett standardformat, så om ett föremål har storlek 3 så får den plats i en låda av volym 3. Om två föremål har storlekarna 1 och 2 så får båda föremålen plats i en låda av storlek 3. Om man lägger dessa föremål i en låda av storlek 4 så finns det fortfarande möjlighet att lägga till ett föremål av storlek 1, men inte större.

Du ska implementera en algoritm för att packa lådor, som går ut på att man itererar igenom föremålen och lägger de i första bästa låda där föremålet får plats.

Skriv funktionen `pack` som tar två parametrar som indata. Den första parametern är en lista som beskriver storleken på ett antal lådor. Listan `[3, 4]` beskriver till exempel två lådor som har volym 3 och 4, respektive. Den andra parametern är en uppslagstabell som där nycklarna är namn på föremål (strängar) och de är associerade med numeriska värden som motsvarare föremålets storlek. Din funktion ska returnera en packning som en lista av listor med föremål som ligger i samma låda. Om alla föremål inte kan packas ska det ges ett särfall (*exception*).

Tester:

```
[In] : pack([1], {'a': 1})
[Out]: [['a']]
[In] : pack([2, 1], {'a': 2, 'b': 1})
[Out]: [['a'], ['b']]
[In] : pack([0, 0], {'a': 1, 'b': 2})
Packing did not work
[In] : pack([1, 3, 1], {'a': 1, 'b': 1, 'c': 1, 'd': 1, 'e': 1})
[Out]: [['a'], ['b', 'c', 'd'], ['e']]
[In] : pack([0, 0, 0], {'a': 1, 'b': 2, 'c': 3, 'd': 1, 'e': 1})
Packing did not work
```

Här ovan har jag minimerat effekten av särfallet till en felutskrift. Utan `try` - `except` blir det förstås mer text.

14. Skriv en funktion `pack_testdata` som slumpar fram testdata till uppgift 13. Givet en lista med lådstorlekar, returnera en uppslagstabell som innehåller föremål med samma storlekar som lådorna, men med ordningen slumpad. Föremålen kan ha godtyckliga namn, t.ex. strängen 'obj' följt av ett unikt tal.

Du får förstås använda modulen `random`.

Tester:

```
[In] : pack_testdata([1])
[Out]: {'obj0': 1}
[In] : pack_testdata([1, 2])
[Out]: {'obj0': 2, 'obj1': 1}
[In] : pack_testdata([2, 1])
[Out]: {'obj0': 1, 'obj1': 2}
[In] : pack_testdata([2, 2, 2])
[Out]: {'obj0': 2, 'obj1': 2, 'obj2': 2}
[In] : pack_testdata([1, 2, 3])
[Out]: {'obj0': 2, 'obj1': 1, 'obj2': 3}
```

Observera att utdata inte behöver se ut exakt som ovan, eftersom slump är inblandat!