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

Tentamen i
DA2005 Programmeringsteknik *och*
DA2004 Programmeringsteknik för matematiker
7.5 hp
2025-08-19

Note: An English version of this exam starts on page 6!

- 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.
- Man måste bli godkänd på del A (4 rätt på 8 frågor) för att del B ska rättas.
- Del B består av frågor med varierande poäng (totalt 12 poäng).
- Inga import (Pythons standardbibliotek eller externa bibliotek) får användas om de inte nämns eller finns med i uppgiften. Man får använda inbyggda funktioner som len, range och map.
- All kod avser **Python 3**, dvs *inte* t.ex. Python 2.7
- **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. Skriv inte lösningarna på tentalydelsen, det tas bort av tentavakten!

1. Vilka av följande alternativ är korrekt Python?

A. `variable = 3`
B. `Variable = 3`
C. `7variable = 3`
D. `variable7 = 3`
E. `_variable = 3`

2. Vad blir resultatet av anropet `f(2)` om funktionerna är definierade som till höger?

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

```
def f(n):
    a = 0
    for i in range(n):
        a = g(a)
    return a

def g(x):
    if x > 10:
        return 1
    else:
        return 2*x + 1
```

3. Vilka påståenden är sanna för Python 3?

A. Uppslagstabeller är muterbara.
B. Uppslagstabeller kan vara nycklar i uppslagstabeller.
C. Uppslagstabeller kan vara värden i uppslagstabeller.
D. Tal (typ int och float) kan vara nycklar i uppslagstabeller.
E. Tal (typ int och float) kan vara värden i uppslagstabeller.

Obs: ”uppslagstabell” avser typen **`dict`**.

4. Vad är resultatet av uttrycket `float('x = 1.0')`?

A. Heltalet `1`
B. Flyttalet `1.0`
C. Variabeln `x` sätts till 1.0 och dess värde returneras
D. Variabeln `x` sätts till 1.0 och `None` returneras
E. Ett särfall

5. Vad innehåller `d` efter att man kört koden till höger?

A. `{ 'A': 0, 'B': 1 }`
B. `{ 'A': 3, 'B': 2 }`
C. `{ 'A': [0, 3], 'B': [1, 2] }`
D. `{ 'A': 0, 'B': 1, 'B': 2, 'A': 3 }`
E. Inget, det blir ett särfall.

```
s = 'ABBA'
d = {}
i = 0
for c in s:
    d[c] = i
    i += 1
```

6. Vilket ord är *inte* associerat med objektorientering?

A. Modul
B. Konstruktor
C. Metod
D. Attribut
E. Klass

7. Vad är resultatet av `list(map(lambda x: (x, -x), range(3)))`?

A. `[0, 1, 2]`
B. `[1, 2, 3]`
C. `[0, 0, 0]`
D. `[-1, -2, -3]`
E. `[(0, 0), (1, -1), (2, -2)]`

8. Vad är resultatet av `['x'* i for i in [1,3,5]]`?

A. `[1, 3]`
B. `[1, 3, 5]`
C. `['x1', 'x3', 'x5']`
D. `['x', 'xxx', 'xxxxx']`
E. Det blir ett särfall.

Del B: kodfrågor

Var snäll använd ett papper till varje fråga i del B. Delfrågor får gärna lösas på samma papper.

9. Skriv funktionen `ai_bot()` som implementerar en *artificiell idiot* ("AI"). (2p)

Funktionen har fyra krav.

- Funktionen ska skriva ut "You: " och vänta på indata.
- På tomt indata ska den svara "Say something." och börja om.
- Om indata är "stop" ska funktionen skriva ut "OK, thanks." och avsluta.
- Annars ska funktionen svara "What?" och börja om.

Exempel på AI-session:

```
[In: ] ai_bot()
You: stop
AI: OK, thanks.
[In: ] ai_bot()
You: hi
AI: What?
You: how are you?
AI: What?
You:
AI: Say something.
You: What is 1+1?
AI: What?
You: stop
AI: OK, thanks.
```

Observera att texten till höger om "You:" antas vara text som användaren matar in.

10. Forskare som studerar arvsmassa (DNA), särskilt hos bakterier, använder ibland begreppet "GC content" för att karakterisera arter eller regioner i arvsmassan. DNA består ju av nukleotider som man brukar representera med bokstäverna A, C, G och T, och man har noterat att andelen G och C kan vara användbar att studera. Vissa grupper av bakterier har till exempel en ganska hög andel G och C, och man har noterat att de delar av arvsmassan där det finns proteinkodande gener är också högre än vad man annars skulle förvänta sig.

A. Skriv funktionen `gc(dna)` som beräknar andelen G och C i den sträng som ges av argumentet `dna`. Du kan anta att indata till funktionen alltid är en sträng över bokstäverna A, C, G och T, och har minst en bokstav. (2p)

B. Bokstaven N används ofta för att markera att vissa positioner eller regioner av DNA är okända. Till exempel är "ACNNGT" en möjlig DNA-sträng där man vet att de två mittersta positionerna finns, men inte vilka av A, C, G eller T som *borde* vara där. Anpassa din lösning så att dessa positioner ignoreras i beräkningen. Du kan anta att det alltid finns minst en av A, C, G eller T i strängen. (1p)

C. Visa hur du kan använda **`assert`** för att skapa ett fel med varningstext om funktionens argument trots allt bryter mot kraven på indata. Det sker om indata är en tom sträng eller inte har annat än N i sig, dvs fall där andel GC inte är definierad. (1p)

Du kan lämna in en lösning som löser alla tre deluppgifterna, om du vill.

Exempel:

```
[In: ] gc('ACGT')
[Out:] 0.5
[In: ] gc('A')
[Out:] 0.0
[In: ] gc('C')
[Out:] 1.0
[In: ] gc('GATTACA')
[Out:] 0.2857142857142857
[In: ] gc('NACNGTN')
[Out:] 0.5
[In: ] gc('')
Traceback (most recent call last):
  File "facit.py", line 98, in gc
    assert n > 0
AssertionError
[In: ] gc('NNN')
Traceback (most recent call last):
  File "facit.py", line 98, in gc
    assert n > 0
AssertionError
```

11. Proteiner är molekyler som kan beskrivas som kedjor av aminosyror. Tack vare kedjestrukturen kan komplicerade molekyler representeras av vanliga strängar över 22 bokstäver som beskriver aminosyrorna (det finns 22 stycken). Till exempel skriver man A för alanin, R för arganin, och N för asparagin, och då kan man tolka ANA som ett protein bestående av alanin, asparagin och alanin i en kort kedja (men riktiga proteiner är större än så).

Man definierar en molekylvikt för aminosyror som mäts i enheten Dalton. Eftersom ett protein är en kedja av aminosyror kan man beräkna proteinets molekylvikt som summan av dess aminosyrors molekylvikt.

Din uppgift är att skriva funktionen `protein_weight(p)` som beräknar och returnerar molekylvikten för det protein som strängen `p` representerar. Funktionen ska summera molekylvikten för aminosyrorna/bokstäverna i `p`. (2p)

Du ska anta att aminosyrornas molekylvikter ges av en uppslagstabell (dictionary) `amino_acid_weight` som ser ut så här:

```
amino_acid_weight = {
    'A':  89.1,
    'R': 174.2,
    'N': 132.1,
    'D': 133.1,
    'C': 121.2,
    # And 15 more
    }
```

Exempel:

```
[In: ] protein_weight('')
[Out:] 0
[In: ] protein_weight('A')
[Out:] 89.1
[In: ] protein_weight('RAD')
[Out:] 396.4
[In: ] protein_weight('ANDRA')
[Out:] 617.6
```

12. Skriv en klass `AminoAcid` för att representera kunskap om aminosyror. Ett `AminoAcid`-objekt ska ha attributen *name* (tex alanine), *abbreviation* (tex A), och *weight* (tex 89.1) (allt skrivet på engelska).

Visa sen hur man skapar ett `AminoAcid`-objekt för alanin. (2p)

13. Skriv funktionen `windowed_gc(dna, k)` som beräknar andel G och C på delsträngar i strängen `dna`, och returnerar en lista med resultatet från funktionen `gc` på alla intervallen. Din lösning ska använda den funktion som definierats i fråga 10 och om du inte har löst den uppgiften kan du ändå anta att funktionen finns. Du ska inte upprepa funktionaliteten från `gc(dna)` i `windowed_gc`. (2p)

Varje position i från `dna` definierar en delsträng och inkluderar k bokstäver före och efter i, om det går. Som exempel, om vi listar delsträngar för olika parametervärden med den positionen som definierar delsträngen i fet stil:

- Om `dna` innehåller ACG och `k=1` så ska tre delsträngar analyseras: **A**C, A**C**G, och C**G**.
- Om `dna` innehåller ACGT och `k=1` så ska fyra delsträngar analyseras: **A**C, A**C**G, C**G**T, och G**T**.
- Om `dna` innehåller ACGT och `k=2` så ska fyra delsträngar analyseras: **A**CG, A**C**GT, AC**G**T, och CG**T**.

Om position i är k bokstäver bort från början eller slutet på strängen ska delsträngen ha längd $2k+1$, eftersom man tar med k bokstäver före position i och k bokstäver efter. När position i är nära början eller slut vill vi ha med så mycket som det går av delsträngarna.

Din funktion ska gå igenom alla positioner från `dna`, extrahera delsträngarna, anropa `gc`, och samla resultaten i en lista som returneras.

Exempel:

```
[In: ] windowed_gc('A', 1)
[Out:] [0.0]
[In: ] windowed_gc('C', 1)
[Out:] [1.0]
[In: ] windowed_gc('ACGT', 1)
[Out:] [0.5, 0.6666666666666666, 0.6666666666666666, 0.5]
[In: ] windowed_gc('AATCGC', 1)
[Out:] [0.0, 0.0, 0.3333333333333333, 0.6666666666666666, 1.0, 1.0]
[In: ] windowed_gc('AATCGC', 2)
[Out:] [0.0, 0.25, 0.4, 0.6, 0.75, 1.0]
```

För den nyfikne: Beräkningar som dessa görs i molekylärbiologin, men då på betydligt större DNA-strängar och med värden på k som är i storleksordningen 50 till 100.

The exam in English

Part A: Multiple choice

Please collect your answers to part A on a single piece of paper. Do not write the solutions on this exam paper, it will be discarded!

1. Which of the following options are valid Python code? (More than one option is possible.)

A. `variable = 3`
B. `Variable = 3`
C. `7variable = 3`
D. `variable7 = 3`
E. `_variable = 3`

2. What will be the result of the call `f(2)` given the function definitions on the right?

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

```
def f(n):
    a = 0
    for i in range(n):
        a = g(a)
    return a

def g(x):
    if x > 10:
        return 1
    else:
        return 2*x + 1
```

3. Which of the following statements are true for Python 3?

A. Dictionaries are mutable.
B. Dictionaries can be keys in dictionaries.
C. Dictionaries can be values in dictionaries.
D. Numbers (of type int and float) can be keys in dictionaries.
E. Numbers (of type int and float) can be values in dictionaries.

4. What is the result of the expression **`float`**`('x = 1.0')`?

A. The integer `1`
B. The float `1.0`
C. The variable `x` is assigned the value 1.0 and its value is returned
D. The variable `x` is assigned the value 1.0 and `None` is returned
E. An exception

5. What is the content of `d` after executing the code on the right?

A. `{ 'A': 0, 'B': 1 }`
B. `{ 'A': 3, 'B': 2 }`
C. `{ 'A': [0, 3], 'B': [1, 2] }`
D. `{ 'A': 0, 'B': 1, 'B': 2, 'A': 3 }`
E. Nothing, an exception is raised.

```
s = 'ABBA'
d = {}
i = 0
for c in s:
    d[c] = i
    i += 1
```

6. Which word is *not* associated with object-orientation?

A. Module
B. Constructor
C. Method
D. Attribute
E. Class

7. What is the result of `list(map(lambda x: (x, -x), range(3)))`?

A. `[0, 1, 2]`
B. `[1, 2, 3]`
C. `[0, 0, 0]`
D. `[-1, -2, -3]`
E. `[(0, 0), (1, -1), (2, -2)]`

8. What is the result of `['x'* i for i in [1,3,5]]`?

A. `[1, 3]`
B. `[1, 3, 5]`
C. `['x1', 'x3', 'x5']`
D. `['x', 'xxx', 'xxxxx']`
E. Nothing, an exception is raised.

Part B: Coding

Please use a separate piece of paper (or several) for each question in part B. Multipart questions such as 9A and 9B can be written on the same piece of paper.

9. Write the function `ai_bot()` that implements an *artificial idiot* ("AI"). (2p)

The function must satisfy four requirements:

- The function should print "You: " and wait for input.
- On empty input, it should respond with "Say something." and start over.
- If the input is "stop", the function should print "OK, thanks." and exit.
- Otherwise, the function should respond with "What?" and start over.

Example AI session:

```
[In: ] ai_bot()
You: stop
AI: OK, thanks.
[In: ] ai_bot()
You: hi
AI: What?
You: how are you?
AI: What?
You:
AI: Say something.
You: What is 1+1?
AI: What?
You: stop
AI: OK, thanks.
```

Note that the text to the right of 'You:' is assumed to be input from the user.

10. Researchers who study genome data (DNA), especially in bacteria, sometimes use the concept of "GC content" to characterize species or regions of the genome. DNA consists of nucleotides, which are commonly represented by the characters A, C, G, and T, and it has been observed that the proportion of G and C can be useful to examine. Certain groups of bacteria, for example, have a relatively high proportion of G and C, and it has also been noted that regions of the genome containing protein-coding genes tend to have higher GC content than would otherwise be expected.

A. Write the function `gc(dna)` that calculates the proportion of G and C in the string given by the argument `dna`. You can assume that the input to the function is always a string consisting of the characters A, C, G, and T, and contains at least one character. (2p)

B. The character N is often used to indicate that certain positions or regions of the DNA are unknown. For example, "ACNNGT" is a possible DNA string where the two middle positions exist but it is unknown which of A, C, G, or T should be there. Adapt your solution so that these positions are ignored in the calculation. You can assume that there is always at least one A, C, G, or T in the string. (1p)

C. Show how you can use **`assert`** to raise an error with a warning message if the function's argument nevertheless violates the input requirements. This happens if the input is an empty string or contains only N's, i.e., cases where the GC content is undefined. (1p)

You may submit a solution that solves all three sub-tasks if you want.

Exempel:

```
[In: ] gc('ACGT')
[Out:] 0.5
[In: ] gc('A')
[Out:] 0.0
[In: ] gc('C')
[Out:] 1.0
[In: ] gc('GATTACA')
[Out:] 0.2857142857142857
[In: ] gc('NACNGTN')
```

```
[Out:] 0.5
[In: ] gc('')
Traceback (most recent call last):
  File "facit.py", line 98, in gc
    assert n > 0
AssertionError
[In: ] gc('NNN')
Traceback (most recent call last):
  File "facit.py", line 98, in gc
    assert n > 0
AssertionError
```

11. Proteins are molecules that can be described as chains of amino acids. Thanks to the chain structure, complex molecules can be represented by ordinary strings over 22 characters that denote the amino acids (there are 22 of them). For example, A stands for alanine, R for arginine, and N for asparagine, so the string ANA can be interpreted as a protein consisting of alanine, asparagine, and alanine in a short chain (although real proteins are larger than this).

A molecular weight is defined for amino acids, measured in units called Daltons. Since a protein is a chain of amino acids, the molecular weight of the protein can be calculated as the sum of the molecular weights of its amino acids.

Your task is to write the function `protein_weight(p)` that calculates and returns the molecular weight of the protein represented by the string `p`. The function should sum the molecular weights of the amino acids/characters in `p`. (2p)

You should assume that the molecular weights of the amino acids are given by `amino_acid_weight`, a dictionary structured as follows:

```
amino_acid_weight = {
    'A':  89.1,
    'R': 174.2,
    'N': 132.1,
    'D': 133.1,
    'C': 121.2,
    # And 15 more
    }
```

Exempel:

```
[In: ] protein_weight('')
[Out:] 0
[In: ] protein_weight('A')
[Out:] 89.1
[In: ] protein_weight('RAD')
[Out:] 396.4
[In: ] protein_weight('ANDRA')
[Out:] 617.6
```

12. Write a class `AminoAcid` to represent knowledge about amino acids. An `AminoAcid` object should have the attributes *name* (e.g., alanine), *abbreviation* (e.g., A), and *weight* (e.g., 89.1) (all in English).

Then show how to create an `AminoAcid` object for alanine. (2p)

13. Write the function `windowed_gc(dna, k)` that calculates the proportion of G and C in substrings of the string `dna`, and returns a list with the results of the function `gc` applied to all such intervals. Your solution should use the function defined in question 10, and if you have not solved that task, you can still assume the function exists. You should not repeat the functionality of `gc(dna)` inside `windowed_gc`. (2p)

Each position i in `dna` defines a substring that includes k characters before and after i, if possible. For example, if we list substrings for different parameter values with the position defining the substring shown in bold:

- If `dna` contains ACG and `k=1`, three substrings should be analyzed: **A**C, A**C**G, and C**G**.
- If `dna` contains ACGT and `k=1`, four substrings should be analyzed: **A**C, A**C**G, C**G**T, and G**T**.
- If `dna` contains ACGT and `k=2`, four substrings should be analyzed: **A**CG, A**C**GT, AC**G**T, and CG**T**.

If position i is fewer than k characters from the start or end of the string, the substring should have length $2k + 1$, since it includes k characters before position i and k characters after. When position i is near the start or end, include as much as possible of the substring.

Your function should iterate over all positions in `dna`, extract the substrings, call `gc` on them, and collect the results in a list that is returned.

Exempel:

```
[In: ] windowed_gc('A', 1)
[Out:] [0.0]
[In: ] windowed_gc('C', 1)
[Out:] [1.0]
[In: ] windowed_gc('ACGT', 1)
[Out:] [0.5, 0.6666666666666666, 0.6666666666666666, 0.5]
[In: ] windowed_gc('AATCGC', 1)
[Out:] [0.0, 0.0, 0.3333333333333333, 0.6666666666666666, 1.0, 1.0]
[In: ] windowed_gc('AATCGC', 2)
[Out:] [0.0, 0.25, 0.4, 0.6, 0.75, 1.0]
```

For the curious: Calculations like these are performed in molecular biology, but then on much larger DNA strings and with values of k on the order of 50 to 100.