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

Tentamensskrivning i
DA2005 Programmeringsteknik
7.5 hp
2026-08-17

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.
- Del A består av flervalsfrågor (totalt 8 poäng).
- Del B består av kodfrågor med varierande poäng (totalt 12 poäng).
- Del C består av kodfrågor inspirerade av labbarna med varierande poäng (totalt 10 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: 15, D: 18, C: 21, B: 24, A: 27, av maximala 30.

Del A: flervalsfrågor (8p)

Var snäll samla svaren på del A på ett svarspapper.

1. Vad skrivs ut av följande kod?

```
print([2*x + 1 for x in range(4)
```

A. `[1, 3, 5]`
B. `[2, 4, 6]`
C. `[1, 3, 5, 7]`
D. `[2, 4, 6, 8]`
E. `[3, 5, 7, 9]`

2. Vad skrivs ut av följande kod?

```
i = 1
while i < 8:
    i *= 2
print(i)
```

A. 4
B. 8
C. 16
D. 32
E. Inget, det blir en oändlig loop.

3. Givet `d={'x': 2, 'y': 5}`, vilket uttryck evaluerar till 5?

A. `d['y']`
B. `d[y]`
C. `d[1]`
D. `d.y`
E. `d.value(5)`

4. Givet definitionen `numbers = [4, 7, 2, 8, 5, 1]`, vad evaluerar `numbers[1::2]` till?

A. `[7]`
B. `[4, 7]`
C. `[7, 2]`
D. `[7, 8, 1]`
E. `[4, 2, 5]`

5. Hur många gånger skrivs ”pling” ut av följande kod?

```
for i in range(0, 10, 6):
    print('pling')
```

A. 1
B. 2
C. 4
D. 6
E. 10

6. Vad skrivs ut av det här uttrycket?

```
values = [1, 2, 3]
print(sum(values), len(values))
```

A. `6 3`
B. `[6, 3]`
C. `(6, 3)`
D. `'sum(values), len(values)'`
E. Inget, det blir en felutskrift

7. Vilka av följande utsagor brukar anses som god praxis för programmering?

A. Skriv hellre en lång funktion istället för flera korta funktioner
B. Ge variabler meningsfulla identifierare.
C. Använd kommentarer för att förklara delar av koden.
D. Undvik att ha tomma rader i koden.
E. Nästla inte loopar och if-satser för djupt.

8. Antag att du vill beräkna $\sin(\pi/8)$. Vilka av följande uttryck gör det?

A.
```
print(sin(pi/8))
```

B.
```
from math import sin, pi
print(math.sin(math.pi/8))
```

C.
```
import math
print(math.sin(math.pi/8))
```

D.
```
from math import sin, pi
print(sin(pi/8))
```

E.
```
import sin, pi
print(sin(pi/8))
```

Del B: kodfrågor (12p)

Var snäll använd ett papper till varje fråga i del B.

9. Betrakta koden här nedan.

A. Förklara med egna ord hur funktionen `mystery` fungerar och vad den returnerar. (1p)
B. Vad skrivs ut av programmet? (1p)

```
def mystery(values):
    result = []

    for x in values:
        if x % 2 == 0:
            result.append(x**2)

    return result

numbers = [3, 4, 7, 2, 5, 8]
print(mystery(numbers))
```

10. Här nedan hittar du en kort funktion som är skriven för att räkna antalet förekomster av ordet givet av variabeln `word` i en textfil given av strängen `filename`. Funktionen har två misstag och din uppgift är att förklara vilka de är. (2p)

```
def count_words(filename, word):
    with filename as f:
        for line in f:
            if word in line:
                n += 1
    return n
```

11. Vi vill beräkna uttrycket `result = 100 / temperature` som del av ett större program. Använd try-except, helt utan if-satser, för att åstadkomma följande: (2p)

- Skriv ut result om beräkningen gick bra.
- Skriv ut ”Temperature cannot be zero” om det blir fel på grund av division med noll.
- Skriv ut ”An error occurred” om något annat särfall uppstår.

12. Betrakta klassen FlightClass här nedan.

A. Skriv kod som instantierar ett objekt för flight ”SK2026” med start i ”ARN” och destination ”UME”. Lagra objektet i variabeln flight. Efter att objektet är skapat ska du sätta status till ”in the air”. (1p)
B. Lägg till en metod `set_as_landed(self)` i klassen. Metoden ska ändra status till ”landed”. Visa sen hur man använder metoden på objektet i variabeln flight. (1p)

```
class FlightClass:
    def __init__(self, flight_number, origin, destination):
        self.flight_number = flight_number
        self.origin = origin
        self.destination = destination
        self.status = 'upcoming'
```

13. Programmerare använder ofta textformatet Markdown för dokumentation och enkla rapporter. Filerna är läsbara som vanlig text, men ser strukturerade ut tack vare enkla markeringar, och det finns verktyg som konverterar dem till Word, PDF med mera.

Som exempel på enkelheten kan vi titta på hur rubriker markeras. En rubrik på nivå 1 (dvs toppnivå) är en textrad som börjar med en ”hash-tagg” följt av mellanslag och själva rubriken. En rubrik på nivå 2 (dvs en underrubrik) har två hash-taggar och naturligtvis har en en rubrik på nivå 3 tre hash-taggar. Här är ett litet exempel med rubriker på nivå 1 och 2:

```
# Programmeringens historia

De första programmen skrevs med hjälp av hålkort.

## Programmering med hålkort

Hålkortens format gjorde att varje rad var exakt 80 tecken.
Det påverkar programmerare än idag.
# De första programspråken
```

Uppgifter:

A. Skriv funktionen `header_level(line)` som tar en sträng som motsvarar en rad i en Markdown-fil och returnerar 0 om det är en vanlig textrad (som ”De första programmen...” ovan) eller nivån på rubriken om strängen börjar med en eller flera hashtaggar. (1p)

B. Du får en extra poäng om `header_level(line)` gör så att det blir ett särfall om det är mer än tre hash-taggar. Rubriker på nivå 4 är alltså inte tillåtna. (1p)

C. Skriv funktionen `table_of_contents(filename)` som öppnar filen `filename` och skriver ut en innehållsförteckning för rubriker på nivå ett och två, och ignorerar rubriker på nivå tre. Rubriker på nivå 2 ska vara indragna två blankslag jämfört med rubriker på nivå 1. (2p)

För enkelhets skull kan du anta att det alltid finns exakt ett mellanslag mellan hash-tagg och rubrik.

Exempelanvändning:

```
  [In: ] header_level('')
  [Out:] 0
  [In: ] header_level('hubba')
  [Out:] 0
  [In: ] header_level('# Programmeringens historia')
  [Out:] 1
  [In: ] header_level('### Tredje nivåns rubrik')
  [Out:] 3
  [In: ] table_of_contents('prog_historia.md') # fil med exempelinnehållet ovan
  [Out:]
Programmeringens historia
  Programmering med hålkort
De första programspråken
  [In: ] header_level('####Nope')
Exception: Too many header levels
```

Del C: kodfrågor inspirerade av labbarna (10p)

Var snäll använd ett papper till varje fråga i del C.

14. Skriv funktionen `inside_unit_circle(x, y)` som avgör (dvs returnerar True eller False) om punkten (x, y) i planet är innanför enhetscirkeln. Du behöver använda modulen math. (2p)

15. Skriv en funktion `deci_to_hexa(num)` som tar in ett tal i det decimala talsystemet och returnerar motsvarande tal i hexadecimal form. (2p)

Om du vill kan du använda den här listan för att slå upp konverteringen av små tal till ett hexadecimalt tal: `lookup = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F']`

Om du tycker att det hjälper kan du också anta att `num` > 0.

Exempel:

```
[In: ] dec_to_hex(1)
[Out:] '1'
[In: ] dec_to_hex(15)
[Out:] 'F'
[In: ] dec_to_hex(16)
[Out:] '10'
[In: ] dec_to_hex(255)
[Out:] 'FF'
[In: ] dec_to_hex(256)
[Out:] '100'
```

16. I laboration 3 implementerade du insättningssortering. En annan klassisk sorteringsalgoritm kallas bubbelsortering (*bubbel sort*) och fungerar så här, på en tänkt indatalista `lst`, om man ska få minsta elementet först.

- Låt `idx` gå från 0 till näst sista index i listan.
- Om `lst[idx]` $>$ `lst[idx+1]` byter man plats på de elementen.
- Upprepa tills att inga element har bytt plats.

Uppgifter:

A. Implementera bubbelsortering i Python. (2p)

B. Hur ser indata ut om man *måste* byta plats på element i varje iteration? (1p)

C. Använd **`assert`** för att skapa ett testfall för din kod, så du automatiskt kan försäkra dig att din implementation är korrekt (åtminstone delvis). (1p)

Exempelanvändning:

```
[In: ] bubble_sort([])
[Out:]  []
[In: ] bubble_sort([2, 1])
[Out:]  [1, 2]
[In: ] bubble_sort([4, 1, 3, 2])
[Out:]  [1, 2, 3, 4]
```

17. Skriv funktionen `plot_my_function` som tar tre argument:

- den funktion som ska ritas (”plottas”),
- startvärde x1 och slutvärde x2.

Vi tänker oss att funktionen, till exempel x^2, ska ritas på intervallet $[x1, x2]$, men bara in heltalspunkterna. Se bilden för ett exempel på utdata.

Man ska kunna anropa funktionen på två sätt, med eller utan skönsvärden (*default*) `x1` och `x2`. Om man utelämnar `x1` och `x2` ska de få värdena -5 och 5.

Använd funktionerna `plot` och `savefig` från Matplotlib (du kan anta att de är importerade) för att göra själva ritningen och se till att utdata blir en PDF. (2p)

English translation

- The exam has multiple-choice questions where at least one answer option is correct. If you answer incorrectly or do not select the exact number of correct alternatives, you receive zero points for the question.
- Part A consists of multiple-choice questions (a total of 8 points).
- Part B consists of coding questions with varying point values (a total of 12 p).
- Part C consists of lab-inspired coding questions, with varying point values (a total of 10 p).
- No **`import`** (Python's standard library or external libraries) may be used unless they are mentioned or included in the assignment. You may use built-in functions such as **`len`**, **`range`**, and **`map`**.
- All code refers to **Python 3**, i.e., *not* e.g. Python 2.7.
- **Allowed aids:** One A4 sheet with as much information as you want. You may write on both sides.
- **Grade thresholds:** E: 15, D: 18, C: 21, B: 24, A: 27, out of a maximum of 30.

Part A: Multiple-choice questions (8p)

Please collect your answers for Part A on a single answer sheet.

1. What is printed by the following code?

```
print([2*x + 1 for x in range(4)
```

A. `[1, 3, 5]`
B. `[2, 4, 6]`
C. `[1, 3, 5, 7]`
D. `[2, 4, 6, 8]`
E. `[3, 5, 7, 9]`

2. What is printed by the following code?

```
i = 1
while i < 8:
    i *= 2
print(i)
```

A. 4
B. 8
C. 16
D. 32
E. Nothing; it becomes an infinite loop.

3. Given `d = {'x': 2, 'y': 5}`, which expression evaluates to 5?

A. `d['y']`
B. `d[y]`
C. `d[1]`
D. `d.y`
E. `d.value(5)`

4. Given the definition `numbers = [4, 7, 2, 8, 5, 1]`, what does `numbers[1::2]` evaluate to?

A. `[7]`
B. `[4, 7]`
C. `[7, 2]`
D. `[7, 8, 1]`
E. `[4, 2, 5]`

5. How many times is "pling" printed by the following code?

```
for i in range(0, 10, 6):
    print('pling')
```

A. 1
B. 2
C. 4
D. 6
E. 10

6. What is printed by this expression?

```
values = [1, 2, 3]
print(sum(values), len(values))
```

A. `6 3`
B. `[6, 3]`
C. `(6, 3)`
D. `'sum(values), len(values)'`
E. Nothing; it produces an error.

7. Which of the following statements are generally considered good programming practice?

A. Prefer one long function over several short functions.
B. Give variables meaningful identifiers.
C. Use comments to explain parts of the code.
D. Avoid blank lines in the code.
E. Do not nest loops and if-statements too deeply.

8. Suppose you want to compute $\sin(\pi/8)$. Which of the following expressions does that?

A.
```
print(sin(pi/8))
```

B.
```
from math import sin, pi
print(math.sin(math.pi/8))
```

C.
```
import math
print(math.sin(math.pi/8))
```

D.
```
from math import sin, pi
print(sin(pi/8))
```

E.
```
import sin, pi
print(sin(pi/8))
```

Part B: Coding questions (12p)

Please use a separate sheet of paper for each question in Part B.

9. Consider the code below.

A. Explain in your own words how the function `mystery` works and what it returns. (1p)

B. What does the program print? (1p)

```
def mystery(values):
    result = []

    for x in values:
        if x % 2 == 0:
            result.append(x**2)

    return result

numbers = [3, 4, 7, 2, 5, 8]
print(mystery(numbers))
```

10. Below you find a short function written to count the number of occurrences of the word given by the variable `word` in a text file given by the string `filename`. The function has two mistakes; your task is to identify and explain them. (2p)

```
def count_words(filename, word):
    with filename as f:
        for line in f:
            if word in line:
                n += 1
    return n
```

11. We want to compute `result = 100 / temperature` as part of a larger program. Using try-except, and without any if-statements, accomplish the following: (2p)

- Print result if the computation succeeded.
- Print ”Temperature cannot be zero” if an error occurs due to division by zero.
- Print ”An error occurred” if any other exception is raised.

12. Consider the class FlightClass below.

A. Write code that instantiates an object for flight ”SK2026” from ”ARN” with destination ”UME”. Store the object in the variable flight. After the object is created, set the status to ”in the air”. (1p)

B. Add a method `set_as_landed(self)` to the class. The method should change the status to ”landed”. Then show how to call the method on the object stored in the variable flight. (1p)

```
class FlightClass:
    def __init__(self, flight_number, origin, destination):
        self.flight_number = flight_number
        self.origin = origin
        self.destination = destination
        self.status = 'upcoming'
```

13. Programmers often use the text format Markdown for documentation and simple reports. The files are readable as plain text but look structured thanks to simple markers, and there are tools that convert them to Word, PDF, and more.

As an example of its simplicity, consider how headings are marked. A level-1 heading (i.e., top-level) is a line of text that starts with a "hash tag" followed by a space and the heading text. A level-2 heading (i.e., a sub-heading) has two hash tags, and naturally a level-3 heading has three hash tags. Here is a small example with level-1 and level-2 headings:

```
# History of Programming

The first programs were written using punched cards.

## Programming with Punched Cards

The punched card format meant that each line was exactly 80 characters.
That still influences programmers today.
# The First Programming Languages
```

Tasks:

A. Write the function `header_level(line)` that takes a string corresponding to a line in a Markdown file and returns 0 if it is a regular text line (such as "The first programs..." above) or the heading level if the string starts with one or more hash tags. (1p)

B. You get an extra point if `header_level(line)` raises an exception when there are more than three hash tags. Level-4 headings are thus not permitted. (1p)

C. Write the function `table_of_contents(filename)` that opens the file `filename` and prints a table of contents for level-1 and level-2 headings, ignoring level-3 headings. Level-2 headings should be indented two spaces compared to level-1 headings. (2p)

For simplicity you may assume there is always exactly one space between the hash tag and the heading text.

Example usage:

```
  [In: ] header_level('')
  [Out:] 0
  [In: ] header_level('hubba')
  [Out:] 0
  [In: ] header_level('# History of Programming')
  [Out:] 1
  [In: ] header_level('### Third level heading')
  [Out:] 3
  [In: ] table_of_contents('prog_history.md') # file with the example content above
  [Out:]
History of Programming
  Programming with Punched Cards
The First Programming Languages
  [In: ] header_level('####Nope')
Exception: Too many header levels
```

Part C: Coding questions inspired by the lab assignments (10p)

Please use a separate sheet of paper for each question in Part C.

14. Write the function `inside_unit_circle(x, y)` that determines (i.e., returns True or False) whether the point (x, y) in the plane is inside the unit circle. You need to use the math module. (2p)

15. Write a function `deci_to_hexa(num)` that takes a number in the decimal system and returns the corresponding number in hexadecimal form. (2p)

If you wish, you may use this list to look up the conversion of small numbers to a hexadecimal digit:
`lookup = ['0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E', 'F']`

If it helps, you may also assume that `num` > 0.

Examples:

```
[In: ] dec_to_hex(1)
[Out:] '1'
[In: ] dec_to_hex(15)
[Out:] 'F'
[In: ] dec_to_hex(16)
[Out:] '10'
[In: ] dec_to_hex(255)
[Out:] 'FF'
[In: ] dec_to_hex(256)
[Out:] '100'
```

16. In lab 3 you implemented insertion sort. Another classic sorting algorithm is called bubble sort and works as follows on a given input list `lst`, sorting with the smallest element first.

- Let `idx` go from 0 to the second-to-last index of the list.
- If `lst[idx]` $>$ `lst[idx+1]`, swap those two elements.
- Repeat until no elements were swapped.

Tasks:

A. Implement bubble sort in Python. (2p)

B. What does the input look like if elements *must* be swapped in every iteration? (1p)

C. Use **`assert`** to create a test case for your code, so you can automatically verify that your implementation is correct (at least partially). (1p)

Example usage:

```
[In: ] bubble_sort([])
[Out:]  []
[In: ] bubble_sort([2, 1])
[Out:]  [1, 2]
[In: ] bubble_sort([4, 1, 3, 2])
[Out:]  [1, 2, 3, 4]
```

17. Write the function `plot_my_function` that takes three arguments:

- the function to be plotted,
- a start value x1 and an end value x2.

The idea is that the function, for example x^2, should be plotted on the interval $[x1, x2]$, but only at integer points. See the figure for an example of the output.

The function should be callable in two ways, with or without default values for `x1` and `x2`. If `x1` and `x2` are omitted, they should default to -5 and 5.

Use the functions `plot` and `savefig` from Matplotlib (you may assume they are imported) to do the actual plotting and ensure the output is saved as a PDF. (2p)