

# CHALMERS

## EXAMINATION / TENTAMEN

Course code/kurskod	Course name/kursnamn			Grade Betyg
TDA548				
Anonymous code Anonym kod		Examination date Tentamensdatum	Number of pages Antal blad	
TDA548-0016-NVL	16 Aug 23	13	5	

\* I confirm that I've no mobile or other similar electronic equipment available during the examination.  
 Jag intygar att jag inte har mobiltelefon eller annan liknande elektronisk utrustning tillgänglig under  
 examinationen.

No/nr	Solved task Behandlade uppgifter	Points per task Poäng på uppgiften	Observe: Areas with bold contour are to completed by the teacher. Anmärkning: Rutor inom bred kontur ifylls av lärare.
1	*	3	
2	*	4	
3	*	5	
4	*	6	
5	*	3	
6	*	6	
7	*	4	<u>Σ31</u>
8	*	4	
9	*	5	
10	*	5	
11	*	4	
12	*	1	
13			
14			
15			
16			
17			
Bonus poäng			
Total examination points Summa poäng på tentamen		50	

<b>CHALMERS</b>	Anonymous code	Points for question (to be filled in by teacher)	Consecutive page no. Löpande sid nr
	Anonym kod <b>T0tS48-0016-NNL</b>	Poäng på uppgiften (ifylltes av lärare)	Question no. Uppgift nr

A1: return är det nyckelord du skriver för att hoppa tillbaka från en metod/funk till resten av din kod. Met. nöjot slags värde. values, referenser eller inget (void).

Return blir vara det sista som görs i metoden/funk, det som kommer efter return exekveras ej.

Exempel:

3

ansver: float = get\_pi()

Jfs getpi() → float:

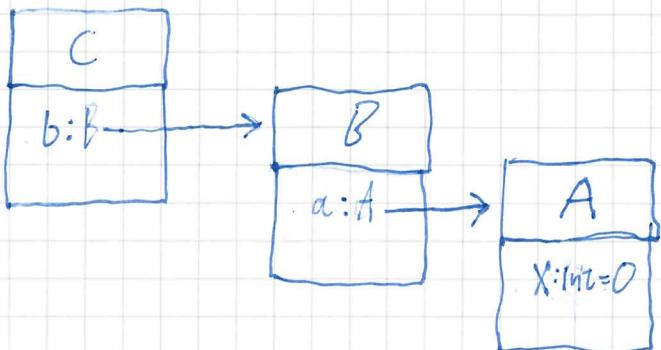
return 3.14

<b>CHALMERS</b>	Anonymous code	Points for question (to be filled in by teacher)	Consecutive page no. Löpande sid nr
	Anonym kod <i>TDA848-0016-NVL</i>	Poäng på uppgiften (ifyller av lärare) <i>4</i>	Question no. Uppgift nr <i>AZ</i>

1.  $x$  assignas 0
2. medan  $x < 12$  loopa följande kod
3. addera 1 till  $x$  ( $x=1$ )
4. om  $x$  felbart med 3 ( $1$  ej felbart med 3)
5. hoppar till else, continue börjar om loopen.
6.  $1 < 12$  loopen fortsätter.
7. addera 1 till  $x$  ( $x=2$ )
8. 2 ej felbart med 3
9. else ger continue
10. addera 1 till  $x$  ( $x=3$ )
11. 3 är felbart med 3
12. gångna  $x$  med 2 ( $x=6$ )
13. går till sista instruktionen. addera 4 ( $x=10$ )
14.  $10 < 12$  så loopen körs
15. addera 1 ( $x=11$ )
16. 11 ej felbart med 3
17. continue
18.  $11 < 12$  fortsätt loopen
19. addera 1 till  $x$  ( $x=12$ )
20. 12 är felbart med 3
21. gångna  $x$  med 2 ( $x=24$ )
22. ihop else eftersom if kördes, hoppar till sista
23. addera  $x$  med 4 ( $x=28$ )
  
24. print( $x$ ) ger 28 *4*

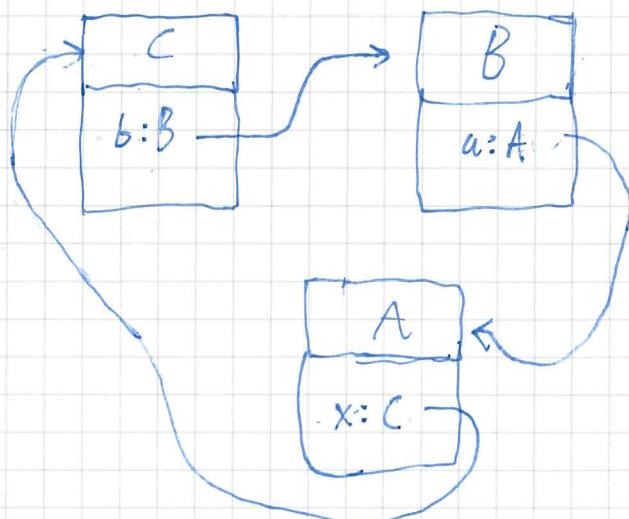
<b>CHALMERS</b>	Anonymous code Anonym kod <i>TDA598-0016-NVL</i>	Points for question (to be filled in by teacher) Poäng på uppgiften (ifyller av lärare)	Consecutive page no. Löpande sid nr 5
			Question no. Uppgift nr A3

Before the call



x?

after the call



Variabler med pythons typ deklaration.

5

<b>CHALMERS</b>	Anonymous code	Points for question (to be filled in by teacher)	Consecutive page no. Löpande sid nr
	Anonym kod TOA 848-0016-NVL	Poäng på uppgiften (ifylltes av lärare)	9 6 Question no. Uppgift nr A4

def validate\_intervals(intervals: list[tuple]):

```

a: int
b: int
old_b: int
for interval in intervals:
    - a = interval[0]
    - b = interval[1]
    - if old_b != None and old_b > a:
        — return False
    - if b < a:
        — return False
    old_b = b
return True

```

Snuggt

Vet inte om sistet intervallet [1,2] finns  
 är också skrivet eftersom det hude ändra typen  
 på listan. Som ges som argument vilket i  
 sig är lite knepigt.

6

<b>CHALMERS</b>	Anonymous code	Points for question (to be filled in by teacher)	Consecutive page no. Löpande sid nr
	Anonym kod TDA548-0016-NVL	Poäng på uppgiften (ifyller av lärare) <b>3</b>	Question no. Uppgift nr <b>A5</b>

En attribut är en variabel som tillhör en klass (klassattribut) och delas av alla dess instanser eller en instans attribut som tillhör en instans av en klass och är "unikt" för just den instansen.

class A:

class\_attrib: int = 5

def \_\_init\_\_(self, arg):  
 self.instance\_attrib = arg

Vi vill använda attributet för att ge en klass eller ett objekt en plats att lagra/entropa relevant information t.ex

person\_obj.age ← instans\_attrib

CarClass.can\_fly ← class\_attrib

**3**

<b>CHALMERS</b>	Anonymous code	Points for question (to be filled in by teacher)	Consecutive page no. Löpande sid nr
	Anonym kod <b>TDA548-001G-NVL</b>	Poäng på uppgiften (ifylls av lärare)	6 Question no. Uppgift nr <b>A6</b>

Exception är något som stör hälften koden

Vi skrivit försöker ge något som inte är möjligt t.ex. dela med 0 eller indexera ett listindex som inte finns.

Vi kan excepta Exceptions i try-except block, i tex mihiväknare är det bra om ZeroDivision error inte kraschar programmet.

Vi kan räisa exceptions om f.dh som användar vär kod ger fel argument till en funktion vi skrivit (Python, språk med dynamic types) Illegal Argument Exceptions.

Vi kanske också vill att exceptions stoppar programmet för att hindra silent bugs. T.ex om en applikation som hämtar internet connection ej får det hela det varit jobbigt om det sätts ut som den fungerade som vanligt för att alla exceptions blev exceptade.

Om man inte på något sätt meddelar att ett fel skett kan det leda till silent bugs, att mycket tror snyggt men gör det ej vilket t.ex kan vara livsamtigt om mycket varan 6 används i typ sjukhus eller rymdväxter. På andra sida kan det också vara lika farligt med mycket varan som kraschar där det hade väckt att bero om ny input.

<b>CHALMERS</b>	Anonymous code	Points for question (to be filled in by teacher)	Consecutive page no. Löpande sid nr
	Anonym kod <b>TDAS48-0016-NVL</b>	Poäng på uppgiften (ifyller av lärare) <b>4</b>	Question no. Uppgift nr <b>A7</b>

En subtyp är en klass som är en del av en annan klass (supertyp) metoder och attribut. Man kan sedan lägga till nya metoder och attribut som är unika till subtypen. I regel syns ej supertypens attribut och metoder i din IDE när du kollar på subtypen.

Vi vill använda subtyper för att undvika kod-duplicering, subtyp polymorphism (alla subtyper kan använda supertypens interface). och sör att det är semantiskt korrekt.

t.ex om vi har superklassen bird med metoden .fly(). Så kan vi behålla alla sarters snygga litudent trots att de är objekt från olika klasser eftersom alla är örvar i sin bird.

Sör att subtyp polymorphism skall fungera är det bra att följa LSP.

"Alla objekt som är subtyper skall kunna ta platsens som dess supertyper har"

dvs ändra inte interface/parametern/returns i det du är örvar.

**4**

<b>CHALMERS</b>	Anonymous code	Points for question (to be filled in by teacher)	Consecutive page no. Löpande sid nr
	Anonym kod <b>TDA548-001G-NNL</b>		8

1. Variabeln r assignas ett R objekt och ger s som arg.
- I R konstruktor kallas supertypen Qs konstruktör. I Q kallas supertypen Ps konstruktör.
- Attributet .a sätts som 5 (argumentet). Sen går vi tillbaka till Qs init a adderas där med 1 ( $a=6$ ). Sen går vi tillbaks till Rs init där a adderas med 3 ( $a=9$ ). ,
2. r.f(7) kallas. I R ser vi att f kallas på supertypens f. Det är nu instansmetoden **Nej**. och inte klassmetoder som kallas (hade kallats som Q.f()). I Qs f(), som vi hoppar till nu multipliceras argumentet med sju självt (a assignas 7\*7).
- Nu är vi tillbaks i R för f()s sluttionen sätter self.a till 53 ( $a=53$ )
3. print r.a ger 53
4. print R.a ger klassattributet 1

4

CHALMERS	Anonymous code Anonym kod TDAS48-0016-NNL	Points for question (to be filled in by teacher) Poäng på uppgiften (ifylls av lärare)	Consecutive page no. Löpande sid nr Question no. Uppgift nr
		5	B2

Ett alias är en variabel som har en referens till samma objekt som en annan variabel.

problem detta kan leda till: Du rörar modifiera ett objekt som du inte vill förändra. Du förändrar något som leder till andra oönskvärda förändringar på alias.

Sätt att du har en funktion sort\_by\_size() som sorterar subklasser av Animal efter Volym. Den tar en lista med objekt och ger tillbaks en lista (sorterad) med objekt. Du använder listan i eat\_the\_biggest() och märker sedan att din Elephant instans är borta. Detta är problem som kan uppstå i exemplet hate en pass-by-value-funktion. Vart styrs (referenser som ges som argument kopieras).

Jag skulle säga att förståelse och objektkopiering styrs dig helt. Vet du att det är samma instans och hur koll på det är alias ett bra verktyg. Vill du inte använda alias?

Gör en deep copy av objektet i sådana fall. Pass by value-funktioner för värdet som används kopieras sanger omkrin.

5

CHALMERS	Anonymous code Anonym kod TOA598-0016-NNL	Points for question (to be filled in by teacher) Poäng på uppgiften (ifyller av lärare)	Consecutive page no. Löpande sid nr <b>10</b>
		<b>5</b>	Question no. Uppgift nr <b>B3</b>

En klass metod är en metod som tillhör klassen (klassobjektet) och har klassen som första argument (Python). En instansmetod är en metod som tillhör instansen av en klass (instansobjekt) och har instansen som första argument. Klass metoden kan därför kallas direkt via klassen istället för ått (som instansmetoden) kallas via instanser av klassen.

Klassmetoden är bra när det inte är logiskt att använda instanser och du alltid vill ha samma operation. Tex

MathematicalOperations.gauss\_eliminate()

Instansmetoden är bra när den helt beror på objektets tillstånd och det är det du önskar.

chicken.move()

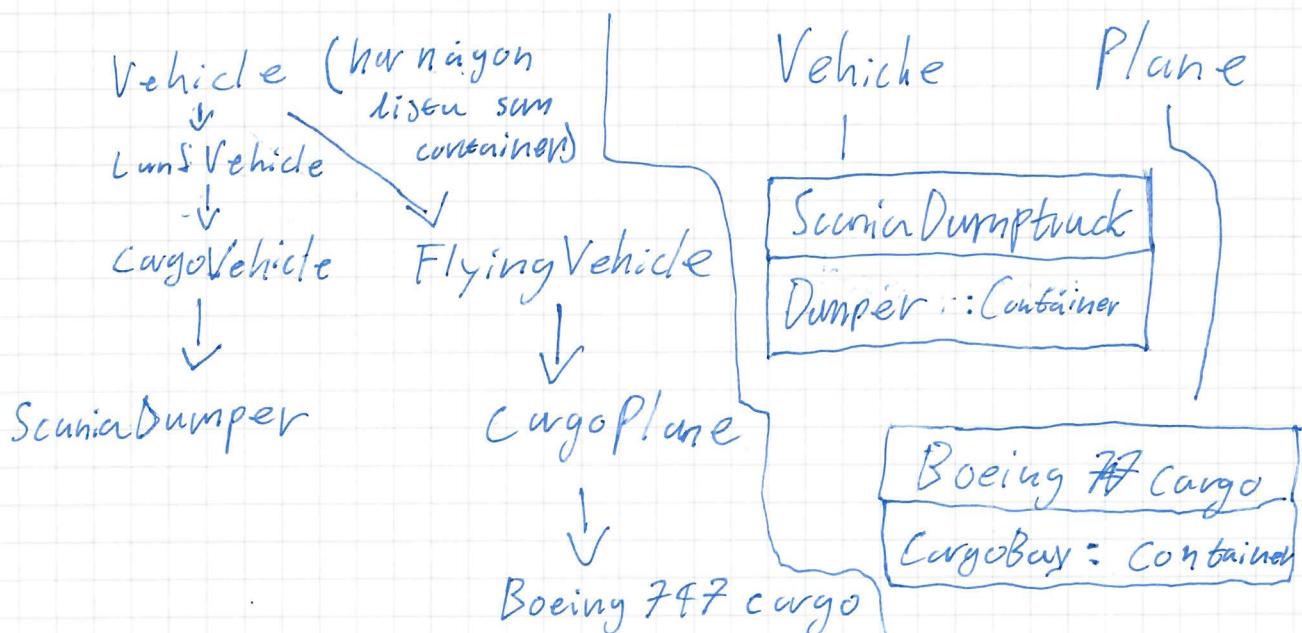
cow.move()

sheep.jump()

**5**

CHALMERS	Anonymous code Anonym kod TDA548-0016-NVL	Points for question (to be filled in by teacher) Poäng på uppgiften (ifylls av lärare)	Consecutive page no. Löpande sid nr Question no. Uppgift nr
----------	---	---	--

Generellt skulle jag säga att det är det som gäller när nägot, semantiskt och funktionellt, är en typ av supertypen t.ex. Ahimals. Kompositionell är här klasser behöver ha samma funktionalitet men kanske annars är helt olika. Det kan dock vara svårt att bestämma i verkligheten och utan kan inte lösa det, betrakt följande exempel:



Sördelen med att det är väl att det blir lite mindre dupliceringsdå man slipper definiera i start sett samma variabel i olika klasser. Nackdelen är att det blir många klasser, man kan också säga sig om FlyingVehicle och LandVehicle borde ha samma supertyp och har lätt det för att implementera utan att bryta mot LSP på något sätt.

<b>CHALMERS</b>	Anonymous code	Points for question (to be filled in by teacher)	Consecutive page no. Löpande sid nr
	Anonym kod	Poäng på uppgiften (ifylltes av lärare)	Question no. Uppgift nr

4

B4

Dessutom kommer att sallet i detta exempel behöva se till att det som läggs till i den gemensanta ärvda containern är rätt saker via dess metoder. Detta hade löst bättre med ett dynamiskt typat container objekt så du kan välja att göra en Container med en Array<Rocks> eller Array<Sand> t.ex.

Detta hade dock också kunnat lösas med Overriding och/eller abstractmethods.

Men i detta exempel hade jag nog valt Composition för att spara tid i utvecklingen.

4

<b>CHALMERS</b>	Anonymous code	Points for question (to be filled in by teacher)	Consecutive page no. Löpande sid nr
	Anonym kod <b>TDAS48-0016-NVL</b>	Poäng på uppgiften (ifylls av lärare)	13 B5

$i = 0$

$j = 0$

$mret = []$

try:

while True:

$m\_copy = copy(m)$

sins ej copy till matrix

Kan man enkelt

vara det med  
för in loopar.

While  $m\_copy.length() > n$

$m\_copy.pop()$

$i += 1$

try:

while True:

$m\_copy_2 = m\_copy.copy()$

for row in  $m\_copy_2$ :

while row.length() > n:

$row.pop()$

$mret.append(m\_copy_2)$

except OutOfIndexError:

$i += 1$

except OutOfIndexError: **WTF?**

return  $mret$