Antal blad / *Number of* sheets: 1 ✓

TENTAMEN / EXAMINATION

Anvisningar: *Skriv din anonymitetskod på varje blad.*
Endast en uppgift får lösas på varje blad.
Var vänlig skriv tydligt!

Instructions: *Write your anonymous code on each sheet.*
Answer only one question on each sheet.
Please write clearly!

Vänligen texta anonymitetskoden i textboxen enligt exempel nedan!
Please write the Anonymous Code clearly in the textbox like example below!

Bokstäver/Letters:
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

Siffror/Numbers:
Ø-1-2-3-4-5-6-~~7~~-8-9

Exempel:

A	B	C	1	~~7~~	Ø	-	Ø	1	~~7~~

DVGA09 Programutveckling och design
Kurskod + Kurs / *Course Code + Course:*

Delkurs / *Part course:*

Anonymitetskod / *Anonymous code* = *Kurskod + kodnr / course code + code number*

D	V	G	A	Ø	9	-	Ø	Ø	4

Tentamensdatum / *Examination date:* 2017-03-31 ✓

Behandlade uppgifter / *Solved problems*

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
X	X	X	X	X	X	X	X							
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30

Ifylles av lärare / *To be completed by the examiner*

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
4	4	5	2,5	5	2,5	4	5							
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30

Poäng / Marks gained: 32
Betyg / Grade: VG
Max poäng / Total marks gained: _____
För Gk poäng / Marks gainedto be passed: _____

Katarina Asplund
Examin. lärare / Kursansvarig signatur / Signature of the examiner

Namnförtydligande / Clarification of the signature

Ange anonymitetskod / *Write your anonymity code*
(Vid icke anonym tentamen ange kurskod + namn + personnummer)
(For non-anonymous exams write the course code + name + civic registration number)

DVGA09-004

Löpande sidnr
Consecutive no:
1

Uppgift nr /
Question no:
1

Poäng / *Points awarded:*

Lärarens anteckning
Examiner's remarks:

1a) Ett gränssnitt är en (klass) som innehåller metodsignaturer. Fördelen med att använda gränssnitt är att klasser som har liknande uppgifter och ska kunna göra liknande saker får enhetliga metodsignaturer. Tex så behöver klasserna "Katt", "Hund" och "Fisk" kanske kunna äta. Metoden för detta skulle då kunna heta eat(). Namnet på metoden förblir då eat() för alla tre klasserna. 2

1b) En abstrakt klass är en beskrivning av hur något kan se ut. Den behöver inte vara skriven i något specifikt språk utan den kan användas i olika programmeringsspråk. Den kan innehålla ofullständiga metoder 0,5 som då blir abstrakta.

1c) Inkapsling sker genom att sätta variabler eller metoder som "private". Det är då bara klassen som de finns i som kan använda dem. Fördelen är att inga andra klasser kan komma åt dem. 1,5

/4

Uppgift nr / *Question no:* 2

Poäng / *Points awarded:*

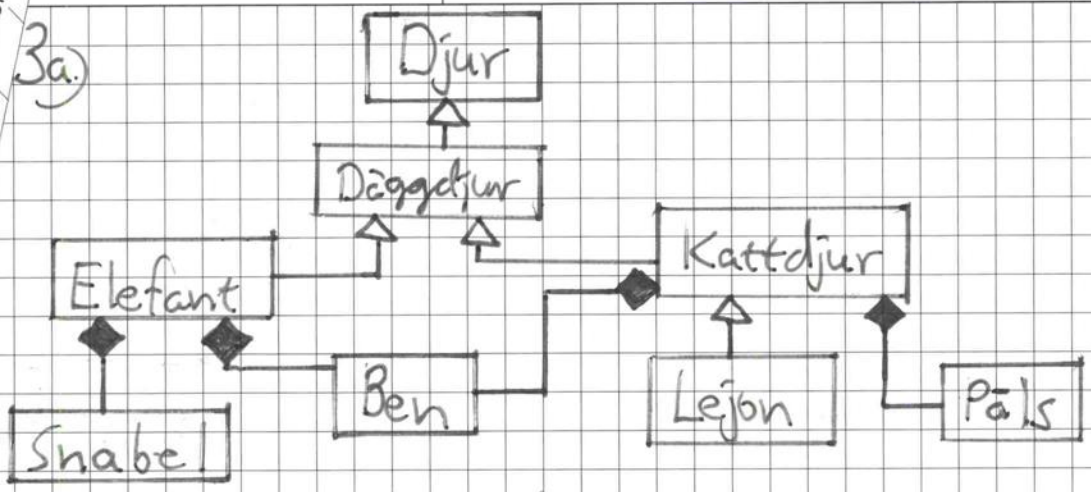
Lärarens anteckning
Examiner's remarks:

2)

```
public class Bankkonto {
private int kontonummer;
private string ägare;
private double saldo;
public Bankkonto(int kont, string owner){
kontonummer = kont;
ägare = owner;
saldo = 0;
}
public void addera Till Saldo (double summa){
saldo = (saldo + summa);
}
public int getKontonummer(){
return kontonummer;
}
public string getÄgare(){
return ägare;
}
public double getSaldo(){
return saldo;
}
}
```

/4

3a)



Om två klasser har relationen arv så innebär det att en av klasserna ärver från den andra. I diagrammet ovan så ser vi att klassen Däggdjur ärver av klassen Djur. Däggdjur ärver då alla variabler och metoder från klassen djur. Klassen Kattdjur ärver från både Däggdjur och Djur. I UML visar man relationen arv som en pil ——▷.
Komposition visar man med symbolen ——◆.
Komposition är en stark relation. Om en klass består av något så är relationen oftast komposition. Om en klass är något så är det arv som gäller. En elefant består av av en snabel, och relationen blir därför komposition. Ett lejon är ett kattdjur och relationen blir då arv.

3

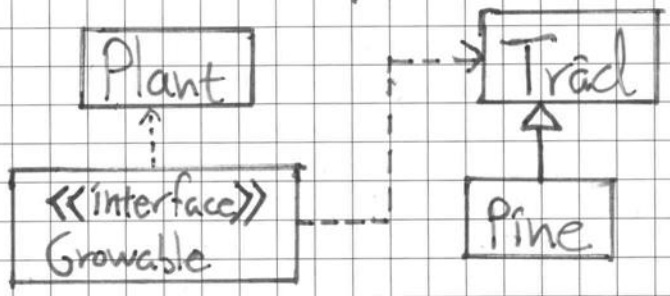
3b.) Jag anser att AAA och BBB har relationen komposition. AAA består av BBB. AAA har en BBB som instansvariabel. Varje gång det skapas ett objekt av AAA så skapas det också ett objekt av BBB.

I klassen AAA finns också en metod där _b, som är en instans av BBB, används.
Det går inte att skapa ett objekt av AAA utan att ett objekt av BBB också skapas.

2

/5

4. Jag ritar ett diagram över klasserna i uppgiften:



Här ser vi vilka relationer klasserna har med varandra.

"p" är av klassen Pine och kan därför använda metoder och variabler från både Pine och Träd. "p" måste också innehålla alla metoder från Growable eftersom Träd implementerar detta gränssnitt.

"träd" är samma sak som "p". "träd" har samma egenskaper som "p". Dock så är "träd" av deklarerade typen Träd.

"växt" är av typen Plant men den får samma egenskaper som "p" eftersom "träd"="p" och "växt"="träd".

"g" deklareras som typen Growable. Men dess faktiska typ är Pine.
Också "g" får samma egenskaper som "p". Detta eftersom Pine är bådas faktiska typ.

12,5

Uppgift nr / *Question no:* 5

Poäng / *Points awarded:*

Lärarens anteckning
Examiner's remarks:

5a) ActionListener är ett interface som måste implementeras i den klass som det ska användas. addActionListener är en metod som man använder när man vill göra något till en "lyssnare". T.ex så kan man göra en JPanel till lyssnare. actionPerformed är den metod man använder när lyssnaren upptäckt att någonting har hänt. Att någon "event" har skett. T.ex att användaren har klickat på en knapp.

1,5

5b.)

```
JButton knapp1 = new JButton();
JButton knapp2 = new JButton();

/* Adderar knapp1 och knapp2 på en
JFrame och gör den till lyssnare */

private void actionPerformed(Event e){
if(e == knapp1) /* Om någon klickade på knapp1 */
   knapp1.doSomething1();
else
   knapp2.doSomething2();
}
```

7

5c.) När man väljer GridLayout så kan man också bestämma utseendet på den. Man väljer hur många rader och kolumner som ens GridLayout ska bestå av. Skickar man t.ex in (3,3) så kommer layouten att se ut så här:

A	B	C
D	E	F
G	H	I

De objekt man lägger till placeras i den ordning de faktiskt läggs till. Det första objektet hamnar i ruta A. Vill man t.ex lägga till en JButton i ruta E så man först lägga till fyra andra komponenter i rutorna A, B, C och D.

2,5

Använder man sig av FlowLayout så placeras komponenter man lägger till automatiskt efter varandra.

1	2	3	4	5

1/5

Använder man BorderLayout så kan man bestämma var en komponent ska hamna genom att skriva CENTER, WEST, NORTH, EAST och SOUTH.

	NORTH	
WEST	CENTER	EAST
	SOUTH	

6. Om felhanteringen inte tar emot ett heltal (integer) så skrivs "Not an Integer, try again" ut.

Om något annat fel sker så skrivs "Something is wrong" ut.

Om de två catch-blocken byter plats så kommer det första blocket alltid att aktiveras eftersom det reagerar på alla sorters fel. Även om ett "InputMismatchException" tas emot så kommer det alltså att fastna i det första blocket ändå. "Something is wrong" kommer då att skrivas ut.

/2,5

Häftområde

Skriv ej i detta område
Leave this area blank

Ange anonymitetskod / *Write your anonymity code*
(Vid icke anonym tentamen ange kurskod + namn + personnummer)
(For non-anonymous exams write the course code + name + civic registration number)

DVGA09-004

Löpande sidnr
Consecutive no:
10

Uppgift nr /
Question no:
7

Poäng / *Points awarded:*

Lärarens anteckning
Examiner's remarks:

7a.) En datastruktur är någonting som implementeras i en abstrakt klass. Det kan t.ex vara en lista, an stack eller ett träd. En datastruktur består av element av en abstrakt datatyp.

0,5

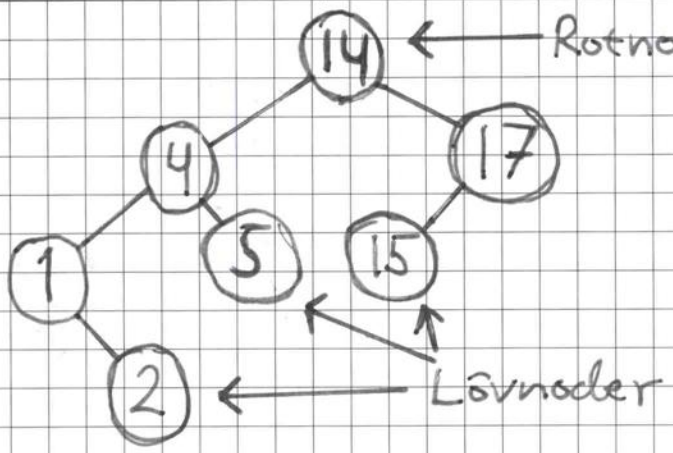
7b.)
```
//pre: true
//post: en tom stack har skapats
public void stack(){
  /* Skapa en stack */
}
//pre true
//post element har lagts till
public void add(E element){
/* Lägg till element i stack */
}
//pre: !isEmpty
//post: element har tagits bort och returnerats
public E remove(){
 /* Ta bort senast tillagda element och
    returnera det */
}

//pre true
//post return true om stacken är tom, annars
       return false.
  public boolean isEmpty(){
  /* Kontrollera om stacken är tom */
  }
```

3,5

/4

8a)



1

8b) Utskrift i postorder: 2, 1, 5, 4, 15, 17, 14

Utskrift i Inorder: 1, 2, 4, 5, 14, 15, 17

2

8c) Rotnod är den första noden man sätter ut, noden högst upp i trädet. I trädet ovan är det (14) som är rotnod.
Lövnoder är de noder som inte har några barn, de noder som är längst ut. I trädet ovan är (2), (5) och (15) lövnoder.

Noddjup är hur många steg ner i trädet noden finns. I trädet ovan så har (2) ett noddjup på 3. Och (17) har ett noddjup på 1.

2

Nodgraden beskriver hur många barn en nod har. I trädet ovan så har (1) ett barn och nodgraden blir därför 1. Även (17) har ett barn och får då en nodgrad på 1.

1/5