SEB BP2 (OOP) week 2 Totaal aantal vragen: 15

De meeste juiste antwoorden: **#1** Minste Juiste antwoorden: **#15**

1. Veronderstel een klasse Beest met een parameter-loze constructor.
Een variabele van die klasse kunnen we als volgt declareren/initialiseren.
(Meerdere antwoorden kunnen goed zijn)

0/11	A	Beest b = Beest();
11/11	B	Beest b = new Beest();
0/11	C	Beest b = new Beest;
0/11	D	new Beest() = b;
9/11	E	Beest b; b = new Beest();

2. Veronderstel weer een klasse Beest met constructor Beest(String naam).
Hoe declareren/initialiseren we een array variabele 'beesten' voor een muis, een kat en een hond?
Meerdere goede antwoorden mogelijk.

9/11	A	Beest[] beesten = {new Beest("muis"), new Beest("kat"), new Beest("hond")};
1/11	B	Beest[] beesten = new Beest[3]; beesten[0] = Beest("muis"); beesten[1] = Beest("kat"); beesten[2] = Beest("hond");
10/11	C	Beest[] beesten = new Beest[3]; beesten[0] = new Beest("muis"); beesten[1] = new Beest("kat"); beesten[2] = new Beest("hond");
0/11	D	Beest[3] beesten = {new Beest("muis"), new Beest("kat"), new Beest("hond")};
0/11	E	Beest[] beesten = {Beest("muis"), Beest("kat"), Beest("hond")};

3. int[] getallen = new int[5];
De variabele getallen is een referentie variabele.

8/11	A	True
3/11	B	False

4. De output van dit processing programma is:
kat
hond
muis

```
class Beest {
  String naam;
  Beest (String naam) {
    this.naam = naam;
  }
}
Beest b1 = new Beest("kat");
Beest b2 = new Beest("hond");
Beest b3 = b1;
b3.naam = "muis";
println(b1.naam);
println(b2.naam);
println(b3.naam);
```

3/11 A True

8/11 B False

5. Veronderstel een klasse Voertuig met 2 constructors.
Voertuig(String naam)
Voertuig(String naam, int max_speed)
Hoe roep je de eerste constructor aan vanuit de tweede?

5/11 A Dat kan niet

4/11 B Voertuig(naam);

1/11 C this().Voertuig(naam);

1/11 D this(naam);

6. Kies het antwoord waar beide beschrijvingen kloppen.
Een geheugenmodel representeert ..... (1) .....
Een klassendiagram representeert ..... (2) .....

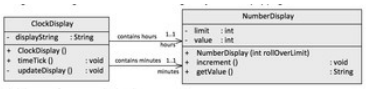
1/11 A (1) hoe het programmageheugen georganiseerd is op een specifiek moment
(2) per klasse de attributen, constructors en methoden

2/11 B (1) een analyse van het programma gedurende zijn uitvoering
(2) per klasse de attributen, constructors, methoden en samenhang met andere klassen

8/11 C (1) hoe het programmageheugen georganiseerd is op een specifiek moment
(2) per klasse de attributen, constructors, methoden en samenhang met andere klassen

0/11 D (1) een analyse van het programma gedurende zijn uitvoering
(2) per klasse de attributen, constructors, methoden en samenhang met andere klassen

7. Hoeveel attributen heeft klasse ClockDisplay?



0/11 A 0

8/11 B 1

0/11 C 2

2/11 D 3

1/11 E 4

8. Wat hoort NIET bij elkaar? (Meerdere goede antwoorden mogelijk)

6/11 A klasse en functie

1/11 B klasse en method

1/11 C klasse en constructor

7/11 D klasse en length

3/11 E klasse en associatie

9. String s1 = new String("Arnhem");
String s2 = new String ("Arnhem");
println (s1 == s2);
Wat is de output?

3/11 A true

7/11 B false

1/11 C syntax fout

10. Veronderstel een klasse Damsteen met 4 public attributen (x, y, diameter, kleur).
Om de damsteen te tekenen kunnen we twee strategieën volgen:
1) een teken methode maken in de klasse Damsteen en deze aanroepen vanuit het hoofdprogramma (b.v. steen.tekenDamsteen())
2) een teken methode in het hoofdprogramma maken die de attributen van Damsteen gebruikt (b.v ellipse(steen.x, steen.y, steen.diameter, steen.diameter);)
Welke strategie geniet de voorkeur en waarom?

1/11 A Strategie 1) omdat dit veel sneller executeert

9/11 B Strategie 1) omdat dit de idee van information hiding beter ondersteunt

0/11 C Strategie 2) omdat het hoofdprogramma zelf wel uitmaakt hoe hij een damsteen tekent

1/11 D Strategie 2) omdat dit een betere encapsulation oplevert

0/11 E Beide strategieën zijn even goed

11. Veronderstel een klasse Dier met twee constructors:
Dier(int i, String s)
Dier (String s, int i)
Deze hebben dus beide een int parameter en String parameter. Mag dat?

9/11 A ja, want de constructors hebben verschillende signatures

2/11 B nee, want de constructors hebben verschillende signatures

0/11 C nee, want de constructors hebben geen return types

0/11 D ja, want de constructors hebben dezelfde signatures

12. Wat is de output van dit programma?

```
class Dier {
  private String naam;
  private int aantalPoten;
  private boolean isLanddier;

  Dier (String naam, int poten, boolean landdier ) {
    this.naam = naam;
    this.aantalPoten = poten;
    this.isLanddier = landdier;
  }

  String toString() {
    return "Dier " + naam + " heeft " + aantalPoten + " poten en is " +
      (isLanddier ? "een" : "geen") + " landdier";
  }
}

Dier vis = new Dier("goudvis", 0, false);
println(vis);
```

0/11 A Dit programma geeft een runtime error.

6/11 B "Dier goudvis heeft 0 poten en is geen landdier"

0/11 C "Vis"

4/11 D Dit programma geeft een compile time error.

0/11 E Geen idee

13. Welke bewering is waar m.b.t. klasse diagrammen? Meerdere goede antwoorden mogelijk.

2/11 A + voor de klasse naam betekent een public klasse
- voor de klasse naam betekent een private klasse

1/11 B + voor een attribuut betekent een private attribuut
- voor een attribuut betekent een public attribuut

8/11 C - voor een attribuut betekent een private attribuut
+ voor een attribuut betekent een public attribuut

6/11 D een pijl van klasse A naar klasse B betekent dat klasse A een attribuut van type B bevat

3/11 E een pijl van klasse A naar klasse B betekent dat klasse B een attribuut van type A bevat

14. Veronderstel een klasse Dobbelsteen en een klasse Dobbelbeker. Klasse Dobbelbeker bevat een attribuut stenen als volgt: Dobbelsteen[] stenen = new Dobbelsteen[3]; Is de volgende bewering juist of onjuist:

Het bijbehorende klasse diagram zal een pijl bevatten van Dobbelbeker naar Dobbelsteen (pijlpunt bij Dobbelsteen). De multipliciteit (3) staat vermeld aan de kant van Dobbelbeker.

4/11 A True

6/11 B False

15. Veronderstel in Eclipse een Java Project met daarin een package P.
P bevat twee klassen: MainProgram en Vierkant.
Vierkant bevat een methode tekenVierkant(). Deze tekent het vierkant m.b.v. de processing functie rect(....) MainProgram heeft een setup() methode. Deze declareert een Vierkant v en roept v.tekenVierkant() aan (zonder parameters).
Het vierkant wordt netjes in een grafisch processing window getekend.

Aan welke voorwaarde moet voldaan zijn om dit goed te laten werken?
Meerdere goede antwoorden mogelijk.

6/11 A Klasse Vierkant moet access modifier *public* hebben.

9/11 B De Referenced Libraries van het Java Project moet de core.jar van processing bevatten.

8/11 C Klasse MainProgram moet processing klasse *PApplet* extenden.

0/11 D Klasse Vierkant mag maximaal 1 constructor hebben.

0/11 E Klasse Vierkant moet een attribuut van type MainProgram bevatten.

6/11 F Klasse MainProgram moet *PApplet.main(...)* aanroepen.

0/11 G De methode tekenVierkant() in klasse Vierkant moet access modifier *private* hebben.