

- het is niet toegestaan een ingeschakelde mobiele telefoon mee te nemen in de tentamenzaal
- het is niet toegestaan de tentamenzaal tussentijds te verlaten voor een w.c. bezoek

Universiteit van Amsterdam
Interfacultair Onderwijsinstituut Informatiewetenschappen
Opleiding Kunstmatige Intelligentie

Tentamen Imperatief & Objectgeoriënteerd Programmeren

Herkansing tentamen deel A+B

**Donderdag 3 juli, 14:00-17:00, zaal P.014,
gebouw Euclides**

Er zijn 30 vragen over de tentamenstof, verdeeld over 3 delen:

- deel 1 :10 begripsvragen van het type "Prelab Activities: Short Answer"
- deel 2: 10 vragen van over gegeven programmacode in Java en C (zie bijlages)
- deel 3: 10 meerkeuzevragen uit de databank met test-items van Deitel & Deitel

Alle vragen zijn van gelijk gewicht bij de beoordeling.

De uitslag is eind volgende week bekend, het tentamen kan eind augustus ingekeken worden.
De uitslag zal op de website van de cursus weergegeven worden.

-tentamen onderdeel 1-

Tien begripsvragen van het type: "Prelab Activities: Short Answer"

Schrijf het genummerde antwoord op het uitgedeelde antwoordformulier.
Geef een ruim antwoord op de gestelde vraag.

- Vraag 1:** Wat wordt bedoeld met het begrip *overloading* van methoden en wat is het verschil met *overriding* van methoden? Geef van beide een duidelijk voorbeeld.
- Vraag 2:** Wat is een *abstract data type* (ADT)? Beschrijf hoe het ADT concept in de programmeertaal Java vorm gegeven wordt.
- Vraag 3:** Wat is het verschil tussen het laten afhandelen van events met behulp van *interfaces* en met behulp van *adapter klassen*?
- Vraag 4:** Wat is het verschil tussen expliciete en impliciete conversie van primitieve typen. Geef een omschrijving en voorbeelden van beide vormen van conversie.
- Vraag 5:** Wat is het verschil tussen een instantie variabele en een klasse variabele? Op welke wijze(n) zijn klasse variabelen toegankelijk?
- Vraag 6:** Wat zijn de overeenkomsten en wat zijn de verschillen tussen gewone methoden, *static* methoden en constructors?
- Vraag 7:** Beschrijf het concept en de werking van *polymorfisme* aan de hand van een eenvoudige, gelaagde klassenstructuur met overerving.
- Vraag 8:** Wat zijn de verschillen tussen gewone klassen (concrete klassen), abstracte klassen en *interfaces*?
- Vraag 9:** Wat wordt bedoeld met het 'catch-or-declare' vereiste bij de afhandeling van exceptions in Java? Op welke wijze zal de programmeur met dit vereiste rekening moeten houden?
- Vraag 10:** Wat zijn de overeenkomsten en verschillen tussen een functie in de taal C en een methode in de taal Java?

-tentamen: onderdeel 2-

Vragen over gegeven programmacode (zie BIJLAGE hfst 3 - 8).

- Zie voor de volgende vraag de bijlage met programmacode:
 - fig. 4.9: **Gradebook.java**
 - fig. 4.10: **GradebookTest.java**

Vraag 11: In de methode `determineClassAverage()` komt een *cast* voor in het if-statement. Wat zijn de gevolgen als we deze *cast* niet opnemen. Levert dit altijd een programmafout op? Zo ja, waarom en welke soort fout (logisch, compilatie, run-time). Zo nee, waarom niet. Geef een toelichting.

- Zie voor de volgende vraag de bijlage met programmacodes:
 - fig. 6.11: **Scope.java**
 - fig. 6.12: **ScopeTest.java**

Vraag 12: Wat wordt bedoeld met het begrip 'shadowing'? Waarom wordt het in de code gebruikt? Waarom ligt het niet voor de hand om het in de methode `useField()` toe te passen?

- Zie voor de volgende vraag de bijlage met programmacode:
 - fig. 7.8: **StudentPoll.java**

Vraag 13: Stel dat we aan de bestaande responses het getal 0 toevoegen, wat verandert er dan aan de uitvoer? Stel dat we aan de bestaande responses het getal 1 toevoegen, wat verandert er dan aan de uitvoer? Stel dat we aan de bestaande responses het getal 11 toevoegen, wat verandert er dan aan de uitvoer?

- Zie voor de volgende vraag de bijlage met programmacode:
 - fig. 7.13: **PassArray.java**

Vraag 14: Stel dat we het eerste statement in de methode `main` vervangen door het volgende:
`int array[] = { 1, 2 };`
Beschrijf het gevolg daarvan voor de compilatie c.q. werking van het programma:

- treden er fouten op, zo ja welke (compilatie, logische, runtime)?
- is er uitvoer, zo ja welke? Beschrijf de uitvoer.

Vragen over gegeven programmacode (zie BIJLAGE hfst 9 - 14).

- Zie voor de volgende vraag de bijlage met programmacodes:
 - **de figuren van hoofdstuk 9, i.h.b. fig. 9.9 t/m 9.11**

Vraag 15: Welke nadelen heeft het gebruik van `protected` instantievariabelen bij overerving? (Deitel & Deitel signaleren drie problemen)

- Zie voor de volgende vraag de bijlage met programmacode: fig. 11.34 **PaintPanel.java**

Vraag 16: In klasse `PaintPanel` wordt een anonieme inner klasse gebruikt als 'event handler'. Welke code wordt daar voor gebruikt? Welke andere conceptuele constructies zijn er in Java om events in een programma af te handelen?

- Zie voor de volgende vraag de bijlage met programmacode: fig. 13.6: **UsingExceptions.java**

Vraag 18: In het programma willen we niet een instantie van `Exception` opwerpen, maar een instantie van de klasse `EigenException`:

```
public class EigenException extends Exception {  
    public EigenException() {  
        System.err.println( "We vangen een EigenException op!" );  
    }  
}
```

Kan dat?, zo ja welke wijzigingen moeten we in het programma aanbrengen?
Kan dat niet? Waarom niet?

- Zie voor de volgende vraag de bijlage met programmacode:
 - fig. 14.17: **AccountRecordSerializable.java**

Vraag 19: In de klassedefinitie van `AccountRecordSerializable` wordt het interface `Serializable` geïmplementeerd. Dit interface wordt een *tagging interface* genoemd, wat wordt daar mee bedoeld en welke eisen stelt het gebruik van dit interface aan de definitie van instantievariabelen?

Vraag over gegeven programmacode in C (zie deze pagina).

o zie bijlage a: **swap.c**

Vraag 20: Is het onderstaande programma swap.c een correct C programma?

- zo ja, wat is de uitvoer?
- zo nee, waar zit de fout?

bijlage a: swap.c

```
#include <stdio.h>
-
void wisselom( int *x_p, int *y_p){
    int *temp;
    *temp = *x_p;
    *x_p = *y_p;
    *y_p = *temp;
}
-
int main(void){
    int getal1, getal2;

    printf("\n tik 2 hele getallen in:\n");
    scanf("%d%d", &getal1, &getal2);
    printf("\n ingetikte getallen: 1:%d 2:%d\n", getal1, getal2);

    wisselom(&getal1, &getal2);

    printf("\n      na omwisseling: 1:%d 2:%d\n", getal1, getal2);
    return 0;
-} //main
```

-tentamen: onderdeel 3-

Tien multiple choice vragen (met één juist antwoord)
uit de testdatabank van Deitel & Deitel

Vraag 21

13.1 Q1: Which of the following statements is not true?

- a. Exception handling enables programmers to write robust and fault-tolerant programs.
- b. Exception handling can only catch the exception but cannot resolve the exception.
- c. Exception handling can resolve exceptions.
- d. The Java 2 Platform, Standard Edition, Version 1.4 introduced the new chained exception feature.

Vraag 22

11.5 Q5: Which of the following answers does not complete the sentence correctly?
An inner class _____.

- a. is frequently used for GUI event handling.
- b. are non-**static**.
- c. does not need a handle to its outer class for methods and variables.
- d. cannot be declared **private**.

Vraag 23

10.1 Q2: Which of the following is not true about interfaces?

- a. An interface describes a set of methods that can be called on an object, providing a default implementation for the methods.
- b. An interface describes a set of methods that can be called on an object, not providing concrete implementation for the methods.
- c. Interfaces are useful when attempting to assign common functionality to possibly unrelated classes.
- d. Once a class implements an interface, all objects of that class have an *is-a* relationship with the interface type.

Vraag 24

13.4 Q4: An uncaught exception:

- a. is a possible exception that never actually occurs during the execution of the program.
- b. is an exception that occurs for which the matching **catch** clause is empty.
- c. is an exception that occurs for which there are no matching **catch** clauses.
- d. is another term for a thrown exception.

Vraag 25

10.2 Q1: For which of the following would polymorphism not provide a clean solution?

- a. A billing program where there is a variety of clients who are billed with different fee structures.
- b. A maintenance log program where a variety of machine data is collected and maintenance schedules are produced for each machine based on the data collected.
- c. A program to compute a 5% savings account interest for a variety of clients.
- d. An IRS program that maintains information on a variety of taxpayers and determines who to audit based on criteria for classes of taxpayers.

Vraag 26

9.2 Q2: An advantage of inheritance is that:

- a. All methods can be inherited.
- b. All instance variables can be uniformly accessed by subclasses and superclasses.
- c. Objects of a subclass can be treated like objects of their superclass.
- d. None of the above.

Vraag 27

9.6 Q2: Which of the following is an example of a functionality that should not be “factored out” to a superclass?

- a. Both ducks and geese are birds that know how to start flying from the water.
- b. All vehicles know how to start and stop.
- c. All animals lay eggs, except for mammals.
- d. All paints have a color.

Vraag 28

5.3 Q2: Consider the classes below:

```
public class TestA
{
    public static void main( String args[] )
    {
        int x = 2, y = 20, counter = 0;

        for ( int j = y % x; j < 100; j += ( y / x ) )
            counter++;
    } // end main
} // end class TestA

public class TestB
{
    public static void main(String args[])
    {
        int counter = 0;

        for ( int j = 10; j > 0; --j )
            ++counter;
    } // end main
} // end class TestB
```

Which of the following statements is true?

- a. The value of counter will be the different at the end of each for loop for each class.
- b. The value of j will be the same for each loop for all iterations
- c. Both (a) and (b) are true.
- d. Neither (a) nor (b) is true.

Vraag 29

6.4 Q1: Variables should be declared as fields only if

- a. they are local variables.
- b. they are used only within a method.
- c. they are required for use in more than one method or their values must be saved between calls to the class's methods.
- d. they are arguments.

Vraag 30

4.7 Q1: What is output by the following Java code segment?

```
int temp;
temp = 180;

while ( temp != 80 )
{
    if ( temp > 90 )
    {
        System.out.print( "This porridge is too hot! " );

        // cool down
        temp = temp - ( temp > 150 ? 100 : 20 );
    } // end if
    else
    {
        if ( temp < 70 )
        {
            System.out.print(
                "This porridge is too cold! ");

            // warm up
            temp = temp + (temp < 50 ? 30 : 20);
        } // end if
    } // end else
} // end while

if ( temp == 80 )
    System.out.println( "This porridge is just right!" );
```

- a. This porridge is too cold! This porridge is just right!
- b. This porridge is too hot! This porridge is just right!
- c. This porridge is just right!
- d. None of the above.

Neem nu de aangekruiste antwoorden over in de tabel op het aparte blad.
Alleen de aangekruiste antwoorden in de tabel worden beoordeeld.