Modulewijzer Web & Mobile: WEM01

Ontwikkel een mobiele applicatie met Java ME

Aantal ECTS
4

Opleiding
Grafimediatechnologie, minor Web & Mobile

Versie
2.1, 2008/2009

A4-Modulebeschrijving

Modulecode:	WEM01
Modulenaam:	Ontwikkel een mobiele applicatie met Java ME
Belasting (aantal CP):	4 CP
Relatie met andere onderwijs	Deze module is een vervolg op de programmeervakken uit eerdere leerjaren en een voorbereiding op WEM04 en WEM05
Voorkennis:	Goede basis in programmeren
Programmasoort	Kennisgestuurd expeditiemodel en studentgestuurd
Werkvormen:	± 1½ uur per week: klassikale bespreking van cases en de achterliggende programmeerconcepten. ± 1½ uur per week: begeleid individueel werken aan opdrachten en bespreken huiswerk ± 8 uur per week: individueel zelfstandig werken aan opdrachten.
Looptijd	10 weken
Toetsing:	Eindproduct (Mobile Game),afsluitend tentamen, en tussentijdse opleveringen
Vrijstelling:	via intake assessment (op basis van aantoonbare ervaringen, product + kennis)
Leermiddelen:	Praktijklokaal. Handboek Objectgeorienteerd programmeren, Jan Beurghs En diverse internet bronnen
Leerdoelen:	De student kent: - Basisbegrippen, termen en het principe van Java / Java ME - De mogelijkheden en beperkingen van het Java ME / mobiele platform - Basisbegrippen, termen en mogelijkheden binnen object georiënteerd denken en Object georiënteerde talen. - De voor- en nadelen van het werken met en denken in objecten - Relevante technieken en schema's voor een object georiënteerd ontwerp in UML - De student kent de mogelijkheden van moderne object georiënteerde talen als Java en C# - De student kent de mogelijkheden en beperkingen van J2ME en het mobiele platform De student kan: - Een casus object georiënteerd analyseren, - Klassen opstellen in Java en omgaan met objecten, - Een spelconcept uitwerken tot gestructureerde ontwerpen (UML) en dit ontwerp realiseren in Java ME
Inhoud:	De twee hoofddoelen van de module zijn leren object georiënteerd denken en werken en het kunnen ontwikkelen in Java ME. Als eindproduct dient een (eigen) gameconcept met behulp van het Java ME platform te worden gerealiseerd. Hiertoe zullen verschillende Java (ME) en object georiënteerde termen en technieken worden behandeld zoals: klassen, objecten, members, data types, casting, encapsulation, inheritance, Threads.
Opmerkingen:	Tijdens de eerste 5 lessen ligt de nadruk op het introduceren van nieuwe kennis. Daarna (of tussendoor) is er ruimte voor verdieping of eventuele herhaling. **Let op**: Aanwezigheid bij eerste 5 lessen is in principe verplicht.
Auteur(s):	M. van Gooswilligen, Mio van der Lijn
Versiedatum:	14 -01-2008
Module beheerder:	Opleiding GMT, Minor Web & Mobile

Inhoudsopgave

1 Algemene omschrijving

1.1 Inleiding

Omdat bijna iedere Nederlander tegenwoordig één of meerdere mobiele apparaten (zoals een mobiele telefoon) bezit en de technische mogelijkheden op dit platform blijven groeien, zijn de verwachtingen voor dit platform hooggespannen. Steeds meer soorten applicaties vinden hun weg naar het mobiele platform. Als gameplatform bijvoorbeeld heeft het andere platforms wat betreft grootte en afzet al voorbijgestreefd.

Hierdoor is de mobiele markt voor jou als media ontwikkelaar een zeer interessante markt om jezelf in te bekwamen. In de rol van media ontwikkelaar zal je daarom met behulp van het **Java ME** platform een mobiele applicatie, een game, ontwikkelen. Java ME (vroeger J2ME) is binnen het mobiele platform momenteel nog steeds marktleider.

Hierbij is specifiek gekozen voor een mobiele Game om een aantal redenen. Niet alleen is gaming (casual met name) één van de snelst groeiende markten, maar game development is ook zeer programmeer-intensief, wat je dwingt tot beter en efficiënter programmeren. Hierbij hoort ook het principe van object georiënteerd programmeren en ontwerpen. Naast dat Java een object georiënteerde taal is, biedt een dergelijke aanpak ook vele voordelen bij de ontwikkeling van deze applicatie, en applicaties in het algemeen.

Zodoende zal er ook ruimschoots aandacht worden besteed aan een object georiënteerde aanpak, zowel in ontwerpfase als realisatie.

1.2 Inhoud

De twee hoofddoelen van de module zijn leren object georiënteerd denken en werken en het kunnen ontwikkelen in Java ME. Als eindproduct dient een (eigen) gameconcept met behulp van het Java ME platform te worden gerealiseerd.

De keuze voor Java ME is vanwege het feit dat deze op dit moment nog steeds marktleider is voor wat betreft software ontwikkeling voor mobiele apparaten. Daarnaast heeft de taal Java vele overeenkomsten met o.a. c# (Microsoft .NET platform) en Actionscript 3.0 (Flash platform) waardoor het gemakkelijker is om eventueel te schakelen naar een ander platform indien de verhoudingen in de markt dusdanig verschuiven. Uiteraard doe je ook programmeerkennis op die altijd van pas zal blijven komen, en is het goed en intuïtief object georiënteerd kunnen denken van onschatbare waarde gedurende je verdere (programmeer) carrière.

Het opzetten van een object georiënteerd spel zal vooral vanuit de praktijk worden belicht. Dat wil zeggen aan de hand van voorbeelden en opdrachten zal je geconfronteerd worden met het hoe en waarom. Wel zal er ruimschoots aandacht zijn voor de toepassing van UML en het vertalen van het ontwerp naar code. Daarnaast is een zeker niveau mbt object georiënteerd denken belangrijk in het kunnen doorgronden van de werking van Java en het Java ME platform.

Om de game te verwezenlijken zullen verschillende Java (ME) en object georiënteerde termen en technieken voorbij komen zoals: Klassen, objecten, members, data types, casting, encapsulation, inheritance en threading. Deze technieken zijn stuk voor stuk direct toepasbaar in je te ontwikkelen spel. Het is daarom belangrijk om bij de lessen aanwezig te zijn en de opdrachten uit te voeren.

1.3 Relatie met andere onderwijseenheden

De module is een vervolg op eerdere modulen mbt programmeren en ontwerpen en gaat uit van een basis kennis daarin. De nadruk ligt op het samen brengen van deze twee aspecten en de relatie duidelijk te maken.
Verder dient de module ter voorbereiding op WEM03, 04 en 05/06 qua programmeerkennis en professionalisering in softwareontwikkeling.

1.4 Leerdoelen

De student kent:

- Basisbegrippen, termen en het principe van Java / Java ME

- De mogelijkheden en beperkingen van het Java ME / mobiele platform
- Basisbegrippen, termen en mogelijkheden binnen object georiënteerd denken en Object georiënteerde talen.
- De voor- en nadelen van het werken met en denken in objecten
- Relevante technieken en schema's voor een object georiënteerd ontwerp in UML
- De student kent de mogelijkheden van moderne object georiënteerde talen als Java en C#
- De student kent de mogelijkheden en beperkingen van J2ME en het mobiele platform

De student kan:
- Een casus object georiënteerd analyseren,
- Klassen opstellen in Java en omgaan met objecten,
- Een spelconcept uitwerken tot gestructureerde ontwerpen (UML)
- dit ontwerp realiseren in Java ME

1.5 Werkvorm(en)

De eerste vijf lessen hebben een grotendeels gelijke invulling. De nadruk ligt hierbij op het overbrengen van nieuwe kennis (of het herhalen hiervan). Dit zal probleem gestuurd worden neergezet, van jou als student wordt dan ook verwacht dat je actief mee doet tijdens de lessen (geen hoorcolleges). De theorie wordt daarom veelvuldig onderbroken/ondersteund door opdrachten die individueel, in groepsverband, of klassikaal worden gemaakt/bediscussieerd.

Elke opdracht draagt in meer of mindere mate bij aan het uiteindelijk kunnen realiseren van je mobiele game en uiteraard aan kennis die nodig is voor het halen van het tentamen. Zeker de lessen 4 en 5 zijn hierbij van groot belang.

De lessen 6 t/m 8(/9) hebben geen ingedeelde theorie maar kunnen worden gebruikt voor herhaling of verdieping. Tijdens de lessen is er uiteraard ook altijd ruimte voor nieuwe vragen met betrekking tot hoe iets op te lossen in je game.

Let op: De ervaring heeft aangetoond dat programmeren alleen te leren is door het heel veel te doen. Het is dan ook belangrijk om vanaf les 1 actief mee te doen, en indien iets niet duidelijk is gelijk om uitleg vragen, elke les is hier ruimte voor aanwezig.
Studenten die voorgaande versies van deze module hebben gevolgd en niet hebben gehaald, zijn zonder uitzondering te laat begonnen met het daadwerkelijk programmeren, of het oppakken van de eindcase.

1.6 Keuzeruimte

De leerdoelen en lesstof van de module liggen vast. De keuze van het spelconcept bepaalt echter in grote mate hoe diep je uiteindelijk het platform in zal gaan. Als je ambitieus bent kan je op deze manier verder in Java ME gaan dan binnen de theorie wordt aangeboden. Denk hierbij aan connectivity met een webserver, multiplayer via bluetooth, of het wegschrijven van 'save' informatie.

1.7 Toetsing

De toetsing bestaat uit een tentamen en het opleveren van het eindproduct (zowel werkend spel, de bijbehorende code, en ontwerpdocumentatie). Het tentamen en de game wegen even zwaar in het eindcijfer, maar de afronding wordt beïnvloed door de mate van aanwezigheid en betrokkenheid in de lessen (en het maken van het huiswerk). Hierdoor kan het zijn dat het eindcijfer een punt lager of hoger uitvalt dan het gemiddelde van de toets en de game.

De ingeleverde game wordt gepresenteerd aan de klas en docent (max 5 minuten). Ook is het mogelijk dat er nog een nabespreking komt over de game. Bijvoorbeeld als de kwaliteit van de game oplevering in grote mate afwijkt van hoe het tentamen is gemaakt. Het is dus niet zo dat iedere student een nabespreking krijgt.

Let op: Aanwezigheid bij de **eerste 5 lessen** is **verplicht** (tenzij vooraf een geldige reden is gegeven).

2 Programma

2.1 welke onderwerpen komen aanbod?

Let op dat de onderwerpen verspreid of gesplitst kunnen zijn over meerdere lessen!

2.1.1 Onderwerp 1: Programmeren /Java algemeen

- Java opbouw: Virtual machine, API
- Programeer/java basisbegrippen:
    - Variabelen
    - Methoden/Functies
    - Control statements (conditioneel, lus, vertakking)
    - Operatoren
    - Structuur (class, package, access modifiers)
    - Arrays & Collections
- OO Programmeren in Java
    - Classes
    - Objecten
    - Instanties
    - Packages

2.1.2 Onderwerp 2: Object georiënteerd denken, ontwerpen en programmeren

- Wat is Object Oriëntatie?
- Objecten, relaties, members
- Interfaces
- Inheritance
- Encapsulation
- Java & OO
- Gaming & OO

2.1.3 Onderwerp 3: UML

- Wat is UML?
- Het gebruiken van UML algemeen en specifiek voor games
- Class diagram
- Activity diagram

2.1.4 Onderwerp 4: Java ME

- Wat is Java ME?
- CLDC/CDC
- MIDlet
- Graphics,
- Interactie
- Resources,
- Packaging

2.1.5 Onderwerp 4: Java (ME) & Threading

- Wat zijn Threads?
- Noodzaak & gebruik
- Multithreading
- Threads, OO & Java

2.2 Te bestuderen literatuur en andere bronnen

Voor het gebruik van OO en UML adviseren we een boek aan te schaffen, zoals het “handboek objectgeorienteerd programmeren” (Nederlandstalig, geniaal eenvoudig, voor een solide basis) of “Obect-oriented Analysis and Design with Applications” (Engelstalig, uitgebreider, ook als referentie te gebruiken). Verder zal in het vervolg van de minor ook regelmatig worden gerefereeerd aan “Praktische UML”.

De meest uitgebreide bron voor Java en Java ME is het internet, en specifiek de site van de makers van java (Sun Microsystems): java.sun.com. Hierbij is o.a. een uitgebreid forum te vinden waar de meeste vragen wel een keer gesteld zijn (zoek ze dus eerst door voordat je zelf hier iets gaan plaatsen!). Hieronder volgt een lijst met url’s naar specifieke onderdelen of tutorials.

Wellicht ten overvloede: Er wordt verwacht dat je zelfstandig voorbeelden en uitleg kan verwerken tot hetgeen je nodig hebt. Gebruik deze links dan ook meer in het kader van “Hoe lees ik een Jpg in en plaats ik deze in mijn Java ME app” dan dat je zoekt naar “Hoe maak ik tetris in Java ME”.

Let op bij het zoeken naar informatie: **Java ME** heette voorheen **J2ME**. Dit kan daarom een betere zoekterm zijn gezien het feit dat veel mensen het nog **J2ME** noemen.

Mocht je toch graag een boek over Java hebben kan je bijvoorbeeld kijken naar de boeken van Deitel & Deitel (niet specifiek Java ME, moeilijk om als startersboek te gebruiken, maar zeer uitgebreid als naslagwerk). Of de boeken van de “Head First” series (ook niet specifiek Java ME, wel interessante manier van informatie overbrengen, maar minder geschikt als naslagwerk).

Feit blijft dat boeken persoonlijk zijn en er weinig echt slechte boeken bestaan. De beste zet is om zelf in de boekhandel te kijken of een boek je aanspreekt, de kans is groot dat als je het prettig vindt lezen, je er ook iets van opsteekt.

Boeken:

- Handboek Objectgeorienteerd programmeren
  Jan Beurghs
  Paperback | 316 Pagina's | Duuren Media, Van
  ISBN10: 9059401174 | **ISBN13:** 978-9059401174
- Object-Oriented Analysis and Design with Applications
  Grady Booch
  Addison-Wesley Professional; 3 edition
  ISBN-10: 020189551X | **ISBN-13:** 978-0201895513
- Praktische UML
  Jos Warmer & Anneke Kleppe
  Paperback | 253 Pagina's | Pearson Education | 4e Editie
  ISBN10: 9043012653 | **ISBN13:** 978-9043012652
- Java How to Program
  (Harvey & Paul) Deitel & associates inc.
  Paperback | 1500 Pagina's | Prentice Hall | 7th edition
  ISBN10: 0132222205 | **ISBN13:** 978-0132222204
- Head First Java
  Kathy Sierra
  Paperback | 720 pages | O’Reilly Media inc. | 2nd edition
  ISBN10: 0596009208 | **ISBN13:** 978-0596009205

Internet:

- Syntax van Java en algemene zaken zoals if/else while etc
  http://java.sun.com/docs/books/tutorial/java/nutsandbolts/index.html

- Basisprincipes van Object Georiënteerd denken/werken.
  http://java.sun.com/docs/books/tutorial/java/concepts/index.html

- Basis implementatie van de objectengedachte in Java, code voorbeelden, etc.
  http://java.sun.com/docs/books/tutorial/java/javaOO/index.html

- tutorial over J2ME:
  http://today.java.net/pub/a/today/2005/02/09/j2me1.html?page=1

- Netbeans IDE tutorials over mobility:
  http://www.netbeans.org/kb/trails/mobility.html

- Java ME overzicht
  http://java.sun.com/javame/technology/index.jsp

- overzicht mobiele telefoons, en de Java ME versie die ze ondersteunen.
  http://developers.sun.com/techtopics/mobility/device/pub/device/list.do?sort=manufacturer&page=7&apiId=61

- API documentatie voor MIDlets
  http://java.sun.com/javame/reference/apis/jsr118/

- J2ME optimalisatie (advanced):
  http://www.microjava.com/articles/techtalk/optimization?content_id=7097

Software/tools:

- Java & Java ME ontwikkelomgeving Netbeans
  http://www.netbeans.org/

- Java & Java ME ontwikkelingomgeving Eclipse
  http://www.easyeclipse.org/site/distributions/index.html

- Visual Paradigm Community Edition (UML tool, freeware)
  http://www.visual-paradigm.com/product/vpuml/vpumldownload.jsp?edition=Community

2.3 Weekschema

Week	Primaire theorie	Praktijk & Producten
1 (7)	Introductie in de les en het einddoel (aantal voorbeelden van games). introductie Java & OO syntax, basisbegrippen zoals variabelen, methods.	Diverse kleine opdrachten ter ondersteuning van de theorie. **Huiswerk**: bestaande uit losse opdrachten java programmeren. + nadenken over eigen gameconcept voor eindopdracht.
2 (8)	(Her)introductie OO principe. Centraal blijft de vraag ”Waarom OO?”. Verder wordt het OO denkproces gekoppeld aan game onderdelen om een relevante en praktische koppeling te tonen. Ook is er een licht begin met UML (diagrammen en hun relatie, meer aandacht voor class diagram) Naast de theorie van OO wordt ook een relatie gelegd naar programmeren. Waarin vooral wordt ingegaan op “ UML -> Code”.	Bespreken huiswerkopdrachten (indien nodig) en koppeling leggen naar theorie van vorige les. Diverse kleine opdrachten ter ondersteuning van de theorie. **Huiswerk**: bestaande uit losse opdrachten OO principes. En 1 opdracht ter voorbereiding op de volgende les/theorie. + nadenken over eigen gameconcept voor eindopdracht.
- (9)	*Voorjaarsvakantie*	
3 (10)	Verder met OO principes en de koppeling naar code. Hoe vertalen we een class diagram en activity diagram naar code?	**Inleveren gameconcept voor eindopdracht.** Bespreken huiswerkopdrachten (indien nodig) en koppeling leggen naar theorie van vorige les. Diverse kleine opdrachten ter ondersteuning van de theorie. **Huiswerk**: Beginnen met **gameconcept** in OO uit te werken (classes, activities).
4 (11)	Introductie Java ME: algemene opbouw, Midlet, user input, graphics.	Bespreken gameconcepten van diverse personen en koppeling leggen naar theorie van vorige les. Diverse kleine opdrachten ter ondersteuning van de theorie. **Huiswerk**: Begin met OO **gameconcept** om te zetten naar Java ME code.
5 (12)	Verdieping Threading in Java ME	Diverse kleine opdrachten ter ondersteuning van de theorie. **Huiswerk**: Game verder ontwikkelen.
6 (13)	*2e paasdag*	**Huiswerk**: Game verder ontwikkelen.
7 (14)	Herhaling en verdieping naar behoefte	**Huiswerk**: Game verder ontwikkelen.
8 (15)	**Presentatie Game**	**Inleveren Game**
9 (16)	**Tentamen**	
10 (17)	Eindevaluatie/beoordeling opgeleverde games en documentatie.	
-	*Meivakantie*	

Over het huiswerk
Huiswerkopdrachten (exclusief gameconcept) hoeven niet te worden ingeleverd maar tijdens de lessen kan worden gevraagd om jouw uitwerking te presenteren/uitwerken voor de klas.

Belangrijke data

- **Week 10:** uiterlijk maandagochtend 9 uur: Gameconcept ingeleverd via mail.
- **Week 15:** uiterlijk maandagochtend 9 uur: Definitieve game + documentatie (1x zip of rar) ingeleverd via mail.
- **Week 15:** tijdens de les: presenteren van jouw game aan de docent en de klas.
- **Week 16:** Tentamen (datum/plaats/tijd nog te bepalen)
- **Week 17**: nabespreking/vraagstelling voor enkele personen, naar behoefte van de docent.

3 Toetsing en beoordeling

3.1 Procedure

Om voor een beoordeling van de module WEM01 in aanmerking te komen moet je aan een aantal criteria hebben voldaan. Hieronder wordt beschreven wat deze criteria zijn:

- *Inleveren producten: Game concept, Mobiele game, Game ontwerpdocumentatie*
  Er geldt een verplichting aan op tijd inleveren van materialen. In het weekschema staat beschreven wat deze data zijn. Hierop kan in de regel geen uitzondering gemaakt worden.
- *Aanwezigheid, betrokkenheid bij lessen 1 t/m 5:*
  Om voor een beoordeling in aanmerking te komen moet je minimaal bij de eerste vijf lessen aanwezig zijn geweest. Uitzonderingen hierop worden alleen gemaakt als dit vooraf aan de lessen is gemeld, en uiteraard met een geldige reden!
- *Huiswerk*:
  Er geldt geen verplichting tot het inleveren van het huiswerk, maar er wordt van uit gegaan dat je zelf de verantwoordelijkheid neemt om oefeningen en opdrachten te bestuderen en uit te voeren. Elke week zal er een herhaling zijn van voorgaande les, en zullen opdrachten worden besproken. Indien je duidelijk laat blijken onvoldoende verantwoordelijkheid te hebben genomen, wordt dit meegenomen in de beoordeling onder de noemer betrokkenheid (zie volgende paragraaf).
- *Tentamen:*
  Tentamen is verplicht.

Het eindcijfer wordt bepaald door het gemiddelde tussen de game en het tentamen (beide 50%). Met eventueel de afronding vanuit de 'betrokkenheid' beoordeling (zie volgende paragraaf).

3.2 Beoordelingscriteria

Voor de beoordeling van het vak en de diverse producten gelden o.a. de volgende criteria:

Gameconcept:

Het gameconcept is je eerste opzet tot het uitwerken van je ontwerpdocumentatie en dus uiteindelijk voor het bouwen van het spel. Hierdoor is het alleen noemen van een bestaand spel zeker niet voldoende. Je zult het spelconcept moeten ontleden in diverse aspecten zoals welke interactie vind er plaats? Wat zijn de precieze spelregels (en wat zijn de randvoorwaarden hier weer bij)? Hoe gebruik ik de invoer en het scherm? Welke stadia bestaan er in mijn spel? Wanneer is het spel voorbij? Wat maakt het spel leuk? Etc etc. Deze informatie gebruik je als uitgangspunt voor het ontwerpdocument.

Game ontwerpdocumentatie:

Net als het uitwerken van een business case voor een reguliere applicatie zal je ook bij je game dezelfde stappen moeten uitvoeren. Zo zal je te maken hebben met het definiëren van de spelregels en de overige eisen die het spelconcept beschrijven. Deze moeten zijn vertaald naar specificaties voor je software systeem. Eventueel kan je spelonderdelen onderverdelen door middel van Use Cases.

Je spel logic is uitgewerkt op basis van de gedefinieerde spelregels uit het concept, middels bijvoorbeeld activity diagrammen of eventueel state-transition diagrammen. Deze zijn vervolgens gebruikt om het klassediagram verder uit te breiden met functies en klassen voor het afhandelen van deze spelregel-logic.

Ook belangrijk is dat het ontwerp niet overbodig is vervuild met zaken die er niet toe doen. Bijvoorbeeld diverse diagrammen die beschrijven hoe de menu structuur werkt zijn waarschijnlijk niet relevant. Uiteindelijk blijft het documenteren, en de UML diagrammen een hulpmiddel om dat wat onduidelijk is (of kan zijn voor anderen) inzichtelijk te maken. Het is dus niet zo dat er minimaal xx diagrammen van type I aanwezig moeten zijn en xx diagrammen van type II. Uiteindelijk maak jij als ontwikkelaar de keuze over wat wel en wat niet moet worden uitgewerkt.

Het is hierbij een goed idee om de volgende stelling in je achterhoofd te houden : "Stel ik moet over een jaar dit spel verbouwen, uitbreiden of aanpassen, weet ik dan nog wat waar gebeurt?".

Mobiele Game

Het project moet in compileerbare versie worden opgeleverd (bij voorkeur een Netbeans project). Indien het spel niet compileert, wordt het geheel als ‘niet ingeleverd’ beschouwd. Daarnaast dient de code een goede reflectie te zijn van je uiteindelijke ontwerp (dus klassen en architectuur komen overeen). Uiteraard heb je nette en goed leesbare code geschreven die waar nodig is voorzien van commentaar. Je hebt je (grotendeels) gehouden aan Java programmeerstandaarden en geen onlogische of niet object georiënteerde code geschreven (of indien wel heb je gedocumenteerd waarom).
Als je al code hebt ‘geleend’ van anderen heb je deze goed geïntegreerd door naamgeving te kiezen die in je applicatie kloppen en geen ongebruikte variabelen over te laten.

Je hebt ook zoveel mogelijk logic die meerdere keren plaats vond verplaatst naar een centrale method of class om de onderhoudbaarheid te vergroten.

Ook heb je rekening gehouden met de usability van het product, zeker gezien de beperkte interface mogelijkheden en schermgrootte zijn de keuzes die je hebt gemaakt steekhoudend.

Betrokkenheid

Als 3e jaars student wordt een professionele houding van je verwacht. Je bent betrokken bij de lessen, en steekt voldoende tijd in de opdrachten. Uiteraard ben je elke les aanwezig, tenzij je een goede reden hebt, en dan communiceer je dat tijdig.

Indien je nadrukkelijk niet betrokken bent zal op basis hiervan het eindresultaat naar beneden worden afgerond. Dit kan uiteindelijk een heel punt schelen op het eindcijfer (zowel positief als negatief, een 5 wordt een 6 of een 6 wordt een 5).

3.3 Voorbeeldtoets

De toets bestaat grotendeels uit multiple choice vragen, aangevuld met enkele open vragen. De vragen variëren van het ontleden van bepaalde code (wat is de uitkomst van…), tot het beantwoorden van vragen met betrekking tot de theorie achter object oriëntatie. Ook zullen er een beperkt aantal vragen gaan over het invullen/tekenen van UML diagrammen (voornamelijk klassendiagram en activity diagram).

Let op: bij het tentamen mogen geen boeken/aantekeningen/etc worden geraadpleegd!

Hieronder twee voorbeelden:

Open vraag: “Wat is het verschil c.q. de relatie tussen een object en een klasse?”

Multiple choice:

Bekijk onderstaande code en geef aan wat de uiteindelijke output zal zijn.

```
public class Tester {
	int var;
	Tester(double var) {
		this.var = (int)var;
	}
	Tester(int var) {
		this("hallo");
	}
	Tester(String s) {
		this();
		System.out.println(s);
	}
	Tester() {
		System.out.println("tot ziens");
	}
	public static void main(String[] args) {
		Tester t = new Tester(5);
	}
}
```

- Er zal niets worden geprint.
- Er wordt “hallo” geprint.
- Er wordt “5” geprint.
- Er wordt “hallo” en daarna “tot ziens” geprint.
- Er wordt “tot ziens” en daarna “hallo” geprint.

3.4 Herkansingen

De **herkansingen** zijn als volgt geregeld:

- bij een onvoldoende resultaat aan het eind van de lesperiode, nadat alle deelcijfers zijn verwerkt tot een eindcijfer, volgt een herkansing in de direct daar opvolgende lesperiode. Dit houdt in dat, nadat het eindcijfer bekend is, de student individuele afspraken maakt met de begeleidende docenten over het te volgen herkansingstraject.

- Voor het tentamen geldt dat deze kan worden herkanst in de volgende periode.

- bij onvoldoend resultaat van het eindproduct worden er individuele afspraken gemaakt met de student over zogenaamde reparatiewerkzaamheden, om dit resultaat om te vormen tot een voldoende resultaat. Dit gebeurt direct nadat beoordeling heeft plaatsgevonden. De reparatie werkzaamheden dienen in de regel uiterlijk 5 weken na het vastleggen van de

reparatieafspraken te zijn ingeleverd, tenzij anders overeengekomen met de docent. Let op dat er extra functionaliteit kan worden verwacht als onderdeel van de reparatieslag.

Indien het totale eindresultaat weer onvoldoende mocht zijn, volgt het cursusjaar daarop een nieuwe kans. Maak je geen gebruik van de herkansingsmogelijkheid, dan vervalt het recht op herkansing. In uitzonderlijke gevallen kan hiervan worden afgeweken, echter uitsluitend in overleg met SLB-coach en begeleidende docenten.