Kristianstad University
SE-291 88 Kristianstad
Sweden
+46 44 20 30 00
www.hkr.se

School of Health and Society
Andreas Nilsson

DA110D - Examination 1
(Practical test)

Allowed aids:
The student is allowed to bring any books, any hand written notes, any computer printed notes, his or her own computer and the student is allowed to use the Internet.

Rules:

1. The examinee shall comply with the invigilator's instructions and take the seat shown by the invigilator
2. The practical test start at the appointed time. Anyone arriving more than 30 minutes late forfeits the right to take the practical test
3. Only in exceptional cases may an examinee borrow aids from other examinees. The invigilator arranges such loans.
4. Cheating is reported and can lead to suspension without a grade. The Vice-Chancellor processes cases of cheating. Decisions on suspension are made by the Disiplinary Board
5. Everybody shall finish the practical test at the end of the practical test time. Extra time is not permitted.
6. The practical test shall be solved individually
7. Students are not allowed to cooperate or communicate (verbally or digitally)

Introduction

During the practical test the student shall individually solve four programming tasks using Java. The tasks will be presented later in this document but first, some practical information will be presented.

All students are encouraged to read the full document before starting to solve the individual tasks listed in the end of the document.

Grading of tasks

Each task is worth 2 points. The points are awarded as follows:

2 points	A task is awarded two points if it is solved and all the listed requirements are fulfilled.
1 point	A task is awarded one point if it is solved but one or more of the listed requirements are not fulfilled.
0 points	No points will be awarded for incomplete tasks or tasks uploaded in the wrong format (see below).

What IDE to use

The student is free to use Eclipse, Netbeans or IntelliJ. Should a problem arise with one of the students installed IDEs during the examination the student is allowed to upload solutions created in different IDEs.

How to organize the code and how to upload the solutions

As mentioned above it is of paramount importance that the student upload his or her solutions in the correct way. Failing to do so will result in zero points awarded for the task uploaded in the wrong way.

Each task shall be solved as a separate project in IntelliJ, Eclipse or Netbeans and the correct way of uploading is one compressed file per task. The compressed file shall contain the complete IDE project.

All uploads shall be done via itslearning and must be done before the end of the practical test session.

The upload link can be found on the course page in the folder Examination 1.

Task 1 (2p)

A travel company needs your help creating a program that can be used to book customers to specific seats in an airplane. In the program it shall only be possible to book one customer per seat.

Create the program according to the following requirements:

1) A customer shall be represented by a class. The class shall store the customers name.

2) The customer class shall take the name as a parameter to the constructor and shall contain a getter method for the name.

3) The airplane shall have 100 seats and shall be represented by an array where the seat number is equal to the index. Each index shall store a customer.

4) The program shall have a menu with three options, ”Add booking”, ”View bookings” and ”Exit”.

5) When adding a booking the program shall ask for the persons name and what seat number to book. The program shall keep asking for a seat number until an empty seat is selected. When both a name and a seat number is selected, a customer object shall be created and stored in the array at the selected seat number.

6) When viewing the bookings a list shall be displayed. The list shall show the seat number and if the seat is occupied or not. If it is not occupied it shall print ”Not booked” otherwise it shall print the name of the customer. An example output:

   ```
   [0] Not booked
   [1] Not booked
   [2] Anna Karlsson
   ```

7) When selecting the exit option in the meny the program shall print ”Thank you! Have a good flight!” and exit the program.

The program must use at least:

- Two classes (main + one other)
- An array
- A loop
- An if-statements or a switch-statement

Task 2 (2p)

A friend of yours want you to create a game for their kid. Your friend want it to be a guessing game similar to the classic board game Battleship (Sänka skepp in Swedish) but as their kid is still young it must be simplified.

Here is a short summary of what the game is about. The game is about trying to find and sink a ship that is randomly placed on a 10x10 board by the computer. It is the users task to try to find the ship by guessing the position of the ship (x and y).

Create the program according to the following requirements:

1) The board shall be represented by a two-dimensional array of booleans.

2) The program shall randomly select a x and y coordinate where the ship shall be placed on the board (by marking that location as true).

3) The selected location shall be printed to the screen for debugging purpose.
   An example output:

```
The ship is placed on (6, 6)
```

4) The user shall keep guessing the location of the ship until the user finds the ship.

5) For each guess the program shall print if it is a ”Hit” or a ”Miss”.
   An example output:

```
New guess:
x = 5
y = 5
Miss!

New guess:
x = 6
y = 6
Hit!
It took 5 guess(es)
```

6) The program shall keep track of how many guesses it takes the user to find the ship. The number of guesses shall be printed when the user finds the ship (See example in point 5 above).

The program must use at least:
- One class (the main class)
- Random or SecureRandom
- An two-dimensional array
- A loop
- An if-statements

Task 3 (2p)

A zoo needs a program to keep track of their animals. This particular zoo only keep two kinds of animals, terrestrial animals (live on land) and aquatic animals (live in water).

Create the program according to the following requirements:

1) All animals have a name so a super class storing the name shall be created.

2) Besides having a name the Terrestrial animal objects stores how many legs they have.

3) Besides having a name the Aquatic animal objects stores how many fins they have.

4) All data stored for the different types of animals (including their name) shall be accessible via getters and setters.

5) The program shall have a menu where the user have four options, ”Add a terrestrial animal”, ”Add a aquatic animal”, ”View all animals” and ”Exit”.

6) Both the menu options to add a new animal shall read the needed data from the user and then create either a Terrestrial animal object or an Aquatic animal object. When created, the object shall be added to an array or an ArrayList.

7) The ”View all animals” option shall print all the animals that has been added. For each printed animal its name and number of legs / fins shall be displayed.

8) The exit option shall close the program.

The program must use at least:

- Four classes (main + three other classes)
- Inheritance (One super class and two sub classes)
- One or more arrays or ArrayLists
- A loop
- An if-statements or a switch-statement

Task 4 (2p)

The company where you work will host a conference and need your help to create the name tags for all the visitors.

Your task is to create a program where the user can enter the name and company of all visitors and then print them all.

Create a program according to the following requirements:

1) The program shall represent each visitor by creating an object from a class that stores the name and company of the visitor. The name and company shall be set by the constructor and the class shall have getters but no setters for the name and company instance variables.

2) The program shall start by asking the user how many visitors to enter.

3) After having read the number of visitors the program shall ask the user to enter the name and company of each visitor. For each visitor an object shall be created and stored in an array or ArrayList.

4) After having created all the visitor objects the program shall automatically print a name tag for each person to the screen. The name tag shall look like the example below:

```
Conference               Karl Persson
(Visitor)                Exam company inc.

Conference               Kajsa Svensson
(Visitor)                Another company
```

   Note that there shall be an empty line between the different visitor name tags.

The program must use at least:

- Two classes (main + one other class)
- An array or ArrayList
- A loop