Name/Nom : .....

Klasse/Classe: .....

Hes·so VALAIS WALLIS
Haute Ecole d'Ingénierie
Hochschule für Ingenieurwissenschaften

TEST FINAL – JAHRESPRÜFUNG

SOLUTION

Informatique 1 | Informatik 1

Anweisung / Consigne :

*Lesen Sie die Fragen gut durch und beantworten Sie diese **leserlich** auf den Aufgabenblättern. Für diese Prüfung dürfen Sie 2 Blätter (mit Vor- und Rückseite) mitnehmen, jedoch keine elektronischen Hilfen.*

Tipp: Verlieren Sie bei einzelnen Fragen nicht zu viel Zeit. Beantworten Sie zuerst die Fragen, die Ihnen keine Probleme stellen, und kommen Sie später auf die für Sie schwierigeren Fragen zurück. Die Skala ist unverbindlich

Lisez attentivement la donnée et répondez de manière **lisible** aux questions. Vous avez droit pour cet examen à un aide-mémoire de 4 pages (2 feuilles recto-verso). Aucun moyen électronique n'est permis.

Un conseil : ne restez pas bloqués sur une question. Répondez tout d'abord aux questions avec lesquelles vous êtes à l'aise et revenez ensuite aux questions posant problème. Le barème indiqué est indicatif.

Question	Points	Bonus	Score
Short questions	6	0	
Understanding code	4	0	
Functions writing	6	0	
Comments extraction in files	6	0	
Delivery schedule	8	0	
Object-Oriented programming	5	0	
Total:	35	0	

This exam has 6 questions, for a total of 35 points.

Rev 0.9 β

Question 1 – Short questions (6 points)

Diese Frage ist in unabhängige Aufgaben unterteilt. Die Anzahl Punkte jeder Aufgabe ist am Rand vermerkt. Cette question est séparée en plusieurs exercices indépendants. Le nombre de points pour chaque exercice est indiqué dans la marge.

[1 Pt] (a) *Was ist eine abstrakte Klasse? Erklären Sie dies zuerst theoretisch und geben Sie anschliessend ein Beispiel.* Qu'est-ce qu'une classe abstraite ? Expliquez théoriquement et donnez un exemple.

> **Solution:** An abstract class is a class which can't be instantiated but is only use in inheritance. For example: a shape is not meant to be instantiated and a triangle inherits from shape.

[1 Pt] (b) *Erklären Sie was ein* layout manager *in Swing ist.*
Expliquez brièvement à quoi sert un *layout manager* en Swing.

> **Solution:** A layout manager is used to arrange and scale the components Swing following rules. For instance, a grid layout will try to arrange the components using a grid with specified dimensions. A flow layout will try to arrange the components one after another.

[1 Pt] (c) *Gegeben Sei* | Soit la fonction

```
1  static int foo(int x, int y) {
2    if(x == 0) {
3      return y;
4    }
5
6    return foo(x-1, y+1);
7  }
```

Was gibt `foo(4,5)` zurück? | Que retourne `foo(4,5)` ?

(c) **9**

[3 Pt] (d) *Wahr oder falsch?* | Vrai ou faux ?

	True	*False*
Das Lesen eines bestimmen n-ten Wertes in einer Liste ist schneller als bei einem Array. La lecture de la n$^{\text{ième}}$ valeur d'une liste est plus rapide que la même opération sur un tableau.	☐	⊗
Eine Klasse kann mehrere Schnittstellen implementieren. Une classe peut implémenter plusieurs interfaces.	⊗	☐
Sind in einer Schnittstelle mehrere Methoden enthalten, kann beim implements ausgesucht werden, welche davon implementiert werden. Lorsqu'une interface possède plusieurs méthodes, on est libre de choisir à l'implémentation lesquelles seront implémentées.	☐	⊗
Die Methode `Integer.max(...)` ist statisch. La méthode `Integer.max(...)` est une méthode statique.	⊗	☐
Ist eine Methode `private`, kann man in abgeleiteten Klassen nicht darauf zugreifen. Sie une méthode est `private`, elle n'est pas accessible dans les sous-classes.	⊗	☐
Ein Algorithmus der Komplexität $\mathcal{O}(n)$ ist für beliebig grosse n immer schneller als ein Algorithmus der Komplexität $\mathcal{O}(log(n))$. Un algorithme de complexité $\mathcal{O}(n)$ est toujours plus rapide qu'un algorithme de complexité $\mathcal{O}(log(n))$, pour toutes les valeurs de n.	☐	⊗

	True	False
Eine abstrakte Methode kann nur in einer abstrakten Klasse existieren. Une méthode abstraite ne peut exister que dans une classe abstraite	⊗	☐
Eine Klasse kann höchstens sechs Konstruktoren aufweisen. On peut avoir au plus six constructeurs dans une classe.	☐	⊗
Code im finally eines `try-catch` wird nur nach dem `try` ausgeführt. Le code dans la partie finally d'un `try-catch` sera uniquement exécuté après le `try`.	☐	⊗
Eine Klasse kann mehrere Methoden mit dem gleichen Namen besitzen. Une classe peut avoir plusieurs fonctions du même nom.	⊗	☐

Question 2 – Understanding code (4 points)

[2 Pt] (a) *Gegeben sei die Methode `what`:* | Soit la méthode `what` décrite ci-dessous:

```
1  public static boolean what(String a) {
2    if (a.length() == 1 || a.length() == 0)
3      return true;
4
5    if (a.charAt(0) == a.charAt(a.length() - 1))
6      return what(a.substring(1, a.length() - 1));
7    else
8      return false;
9  }
```

*Geben Sie den Rückgabewert **und die Anzahl der rekursiven Aufrufe** der Methode für:*
Veuillez donner la valeur retournée **et le nombre d'appels récursifs** de la méthode pour:

1) `what("ab")`

1) **false, 0x**

2) `what("abca")`

2) **false, 1x**

3) `what("radar")`

3) **true, 2x**

4) `what("nellen")`

4) **true, 3x**

[2 Pt] (b) *Gegeben sei* | Soit

```
1  static public String stegano2(String inFile) {
2    Vector<String> lines = new Vector<String>();
3    try {
4      BufferedReader bf = new BufferedReader(new FileReader(inFile));
5
6      do{
7        String content = bf.readLine();
8        if(content == null) break;
9        lines.add(0, content);
10     } while(true);
11
12     bf.close();
13   }
14   catch(Exception e) {
15     e.printStackTrace();
16   }
17
18   String r = ""; int i = 1;
19
20   for(String s : lines) {
21     r += s.charAt(s.length() - s.length() + i++);
22   }
23
24   return r;
25 }
```

Was zeigt der Code an, falls die Datei mit dem Namen `fn` *folgenden Inhalt hat?*
Qu'affiche ce code si le fichier correspondant à `fn` contient le texte suivant ?

```
asdfwrejkrxzcuvw
ouiiexcmkjaskdjw
iqupicbvjkaopqoi
weuoirzuiikjyhky
ashdfuyxczuiwqui
```

Solution: `super`

Question 3 – Functions writing (6 points)

[2 Pt] (a) *Schreiben Sie eine Funktion* `times`*, die eine eingegebene Zeichenkette* `n` *mal wiederholt. Zum Beispiel:*
On vous demande d'écrire une fonction `times` permettant de répéter `n` fois un `String`. Par exemple:

```
1 times("*", 5); // returns "*****"
2 times("hello", 3); // returns "hellohellohello"
```

Die Funktion soll das mit jeder x-beliebigen Zeichenkette am Eingang ausführen.
Cette fonction doit pouvoir fonctionner avec n'importe quel `String` en entrée.

Solution:

```
1 public static String times(String s, int n) {
2   if(n <= 0)
3     return "";
4   else
5     return s + times(s, n-1);
6 }
```


⚠*Turn page* →

[4 Pt] (b) *Schreiben Sei eine Funktion, die zwei Vektoren mit ganzen Zahlen als Parameter annimmt und die einen Vektor mit ganzen Zahlen zurückgibt, welches die Elemente der beiden eingegebenen Vektoren enthält, jedoch in zufälliger Reihenfolge.*

Achten Sie darauf, dass ihr Ergebnis die Eingabereihenfolge nicht lediglich übernimmt und dass die beiden Arrays der Eingabe nicht unbedingt die gleiche Anzahl Elemente haben. Beispiel: mit `a` *gleich* `1,2,3` *und* `b` *gleich* `4,5,6`*, dann* `shuffle(a, b)` *könnte* `4,3,2,6,1,5` *oder* `6,2,3,1,5,4` *zurückgeben.*

Hinweis*: Erstellen Sie zunächst einen (Ergebnis-)Vektor, in welchem Sie zuerst die Elemente des ersten Vektors und im Anschluss die des zweiten Vektors speichern und mischen Sie danach die Element des Ergebnisvektors.*

Écrivez une fonction recevant deux vecteurs d'entiers en entrée et qui retourne un vecteur d'entiers en sortie. Celui-ci doit contenir les éléments des deux vecteurs en entrée, mais mélangés de manière aléatoire.

Par exemple, si `a` contient les valeurs `1,2,3` et `b` les valeurs `4,5,6`, alors `shuffle(a, b)` pourrait retourner `4,3,2,6,1,5` ou encore `6,2,3,1,5,4`. Faites attention que votre mélange ne prenne pas uniquement les valeurs dans l'ordre. Les deux vecteurs n'ont pas forcément la même taille !

Conseil : faites d'abord un vecteur contenant les deux vecteurs l'un derrière l'autre puis enlevez des éléments de ce vecteur de manière aléatoire pour les mettre dans le vecteur résultat.

Einige nützliche Methoden der Klasse `Vektor`*:*
Quelques méthodes utiles de la classe `Vector`:

```
1  // Appends the specified element e to the end of this Vector of type E
2  add(E e)
3
4  // Inserts the specified element of type E at the specified position in this Vector.
5  add(int index, E element)
6
7  // Appends all of the elements in the specified Collection to the end of this Vector.
8  addAll(Vector<E> c)
9
10 // Returns the element of type E at the specified position in this Vector.
11 E get(int index)
12
13 // Removes the element of type E at the specified position in this Vector and returns it.
14 E remove(int index)
```

Solution:

```java
public static Vector<Integer> shuffle(Vector<Integer> a, Vector<Integer> b){
  Vector<Integer> res = new Vector<Integer>();
  Vector<Integer> sum = new Vector<Integer>();

  sum.addAll(a);
  sum.addAll(b);

  Random rnd = new Random();

  while(!sum.isEmpty()) {
    int index = rnd.nextInt(sum.size());
    res.add(sum.remove(index));
  }

  return res;
}
```

Question 4 – Comments extraction in files (6 points)

Gegeben sei eine Datei, die Java-Code enthält. Sie möchten die folgende Funktion implementieren:
Soit un fichier qui contient du code Java. On souhaite écrire la fonction :

```
1  static void extractComments(String in, String out)
```

Diese Funktion soll in die durch `out` *bezeichnete Datei alle einzeiligen Kommentarzeilen der durch in bezeichnete Datei schreiben. Zur Erinnerung: dabei handelt es sich um die Zeilen, die mit einem // beginnen.* ⚠ *Die Kommentare beginnen nicht unbedingt in Spalte 1!*

Cette fonction va écrire dans le fichier dont le chemin est donné par `out` toutes les lignes du fichier de code (dont le chemin est `in`) contenant des commentaires sur une ligne. Pour rappel, ceux-ci sont introduits par `//` et il n'y a qu'un seul commentaire par ligne. ⚠ Les commentaires ne commencent pas toujours au début de la ligne.

Beispieldatei für `in` | Par exemple, pour le fichier correspondant à *in* suivant:

```
1  package exams.final2020;
2
3  // This is a commented class
4  public class Comment {
5    double bar; String hello;
6    int foo; // Here is a variable;
7
8    // The main method of our code
9    public static void main(String[] args) {
10     Comment c = new Comment();
11     c.bar = 1.0; // Assignation of a variable
12   }
13 }
```

resultiert in der folgenden Ausgabe für `out` | Le fichier *out* sera

```
// This is a commented class
// Here is a variable;
// The main method of our code
// Assignation of a variable
```

Solution:

```java
public static void extractComment(String in, String out) {
  String outputContent = "";

  try {
    BufferedReader bf = new BufferedReader(new FileReader(in));

    String line = bf.readLine();
    while(line != null) {
      int index = line.indexOf("//");

      if(index != -1) {
        outputContent += line.substring(index) + "\n";
      }

      line = bf.readLine();
    }

    FileWriter fw = new FileWriter(out);
    fw.write(outputContent);
    fw.close();
  }
  catch(Exception e) {
    e.printStackTrace();
  }
}
```

Question 5 – Delivery schedule (8 points)

Um ein Heimliefergeschäft zu gründen, entscheiden Sie ein Planungstool zu programmieren. Sie beginnen mit den beiden folgenden Klassen:

Pour pouvoir lancer votre entreprise de livraison à domicile, vous décidez d'écrire un programme de planification. Vous partez des deux classes suivantes :

```
1  class Customer {
2    String name; String address;
3
4    Customer(String name, String address){
5      this.name = name; this.address = address;
6    }
7  }
8
9  class Order{
10   Customer c;
11   int duration;
12
13   Order(Customer c, int duration){
14     this.c = c; this.duration = duration;
15   }
16
17   public String toString() {
18     return duration + " min for " + c.name;
19   }
20 }
```

Darauf basierend erstellen Sie ein Array mit 8 Positionen, jede korrespondiert mit einer ganzen Stunde des Tages. Der Index 0 des Arrays entspricht dem Zeitfenster 8h-9h am Morgen und die nachfolgenden Indizes bis 7 die Stunde zwischen 15h und 16h. An jeder Position des Arrays speichern Sie einen Vektor mit Bestellungen (`Vector<Order>`) für die entsprechende Periode. Dieser Vektor enthält die Bestellungen `Order`, jede mit der Dauer einer Lieferung sowie den Kundeninformationen. Der entsprechende Code ist gegeben:

À partir de cela, vous construisez un tableau de 8 positions, chacune correspondant à une heure complète de la journée. L'indice 0 du tableau correspond aux horaires entre 8 et 9 heure du matin et ainsi de suite jusqu'à l'indice 7 pour les heures entre 15h et 16h. A l'intérieur de ce tableau, pour chaque position, vous stockez un vecteur de commandes à livrer `Vector<Order>` durant cette période. Ce vecteur contient des commandes (`Order`), chacune ayant une durée de livraison ainsi que les informations du client. Le code correspondant est le suivant :

```
1  public class ReservationStation {
2    Vector<Order> schedule[]= new Vector[8];
3
4    ReservationStation() {
5      for(int i = 0; i < schedule.length; i++) {
6        schedule[i] = new Vector<Order>();
7      }
8    }
9
10   boolean availableInSlot(Order o, int slot) {
11     return timeLeft(slot) >= o.duration;
12   }
13
14   void scheduleOrder(Order o, int slot) {
15     if(availableInSlot(o, slot)) {
16       schedule[slot].add(o);
17     }
18   }
19
20   // Returns how many minutes are still available in a given slot
21   int timeLeft(int slot_in_schedule) {...}
22 }
```

Das folgende Beispiel zeigt die Verwendung des Codes (z.B: innerhalb des `main()`).

On peut utiliser ce code ainsi (par exemple dans le `main`).

```
ReservationStation r = new ReservationStation();

Customer c1 = new Customer("Tony Stark", "Sunset Blvd");
Customer c2 = new Customer("Wanda Maximoff", "Pine Street");
Customer c3 = new Customer("Stephen Strange", "Crescent Road");
Customer c4 = new Customer("Bruce Banner", "Gamma Ray drive");

Order o1 = new Order(c1, 20); Order o2 = new Order(c2, 50);
Order o3 = new Order(c3, 30); Order o4 = new Order(c4, 15);

r.scheduleOrder(o1, 0); r.scheduleOrder(o3, 0);
r.scheduleOrder(o2, 3); r.scheduleOrder(o4, 1);
```

[2 Pt] (a) *Implementieren Sie die Methode `timeLeft()`, die zurückgibt, wieviel Zeit in einem Zeitslot verbleibt*

Donnez l'implémentation de la méthode `timeLeft` qui retourne combien de temps est encore disponible dans un slot donné

Solution:

```
int timeLeft(int slot_in_schedule) {
  Vector<Order> s = schedule[slot_in_schedule];
  int timeLeft = 60;
  for (Order o : s) {
    timeLeft -= o.duration;
  }
  return timeLeft;
}
```

[3 Pt] (b) *Implementieren Sie die Methode `toString()`, do dass `System.out.println(r)` exakt das Ergebnis gemäss folgenden Beispiel anzeigt (berücksichtigen Sie genau die angegebenen Zwischenräume.*

Donnez l'implémentation de la méthode `toString` pour que `System.out.println(r)` affiche exactement, pour l'exemple ci-dessus, le résultat suivant sur la console (attention aux détails d'espaces etc…).

```
Schedule for today:
 8:00/9:00 --> 20 min for Tony Stark; 30 min for Stephen Strange;
 9:00/10:00 --> 15 min for Bruce Banner;
 10:00/11:00 --> Nothing
 11:00/12:00 --> 50 min for Wanda Maximoff;
 12:00/13:00 --> Nothing
 13:00/14:00 --> Nothing
 14:00/15:00 --> Nothing
 15:00/16:00 --> Nothing
```

Solution:

```
1  public String toString() {
2    String r = "Schedule for today:\n";
3
4    for (int i = 0; i < schedule.length; i++) {
5      int start = 8 + i;
6      int end = 8 + i + 1;
7
8      r += " " + start + ":00/" + end + ":00 --> ";
9
10     if (schedule[i].isEmpty())
11       r += "Nothing";
12     else {
13       for (Order o : schedule[i]) {
14         r += o + "; ";
15       }
16     }
17
18     r += "\n";
19   }
20   return r;
21 }
```

[3 Pt] (c) *Implementieren Sie die Methode* `Vector<Integer> availableSlots(Order o)`*, die eine Zusammenstellung möglicher Zeitslots (zwischen 0 und 7) zurückgibt, für welche die als Parameter übergebene Bestellung eingeplant werden kann. Verwenden Sie dabei die bereits implementierten Methoden. Z.B gibt* `availableSlots(new Order(c4, 15))` *den Vektor* `[1,2,4,5,6,7]` *zurück, da es möglich ist, einen Bestellung von 15min in den Zeitslots 9h-10h, 10h-11h, . . . 15h-16h Einzuplanen.*

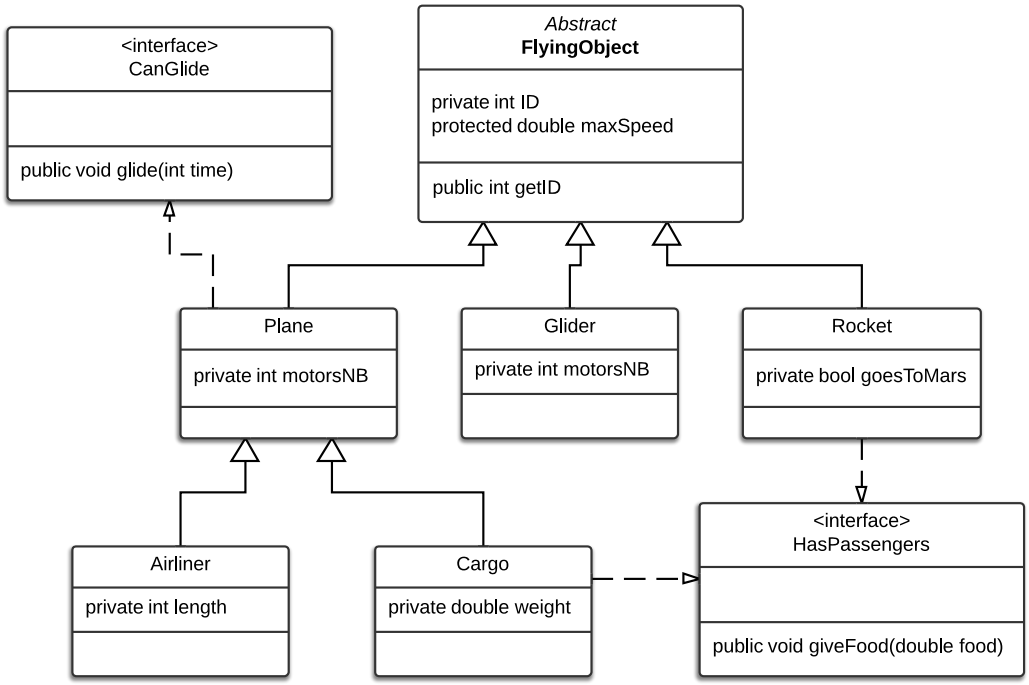
Donnez l'implémentation de la méthode `Vector<Integer> availableSlots(Order o)` qui retourne l'ensemble des slots de la journée (entre 0 et 7) pour lesquels la commande passée en argument est faisable (utilisez les méthodes existantes). Par exemple, `availableSlots(new Order(c4, 15))` retournera le vecteur `[1,2,4,5,6,7]` car il est possible de mettre une commande de 15 minutes aux horaires 9h-10h, 10h-11h, . . . 15h-16h.

Solution:

```
1  Vector<Integer> availableSlots(Order o) {
2    Vector<Integer> v = new Vector<Integer>();
3
4    for (int i = 0; i < schedule.length; i++) {
5      if (availableInSlot(o, i)) {
6        v.add(i);
7      }
8    }
9
10   return v;
11 }
```

Question 6 – Object-Oriented programming ✈ (5 points)

Gegeben sei die folgendes UML Diagram. Beantworten Sie die folgenden Fragen:
Soit la représentation UML suivante. Répondez aux questions suivantes:



Kann man eine Methode giveJournals zur Schnittstelle `HasPassengers` *hinzufügen?*
Pourrait-on ajouter une méthode `giveJournals` à l'interface `HasPassengers`?

True	*False*
⊗	☐

Implementiert die Klasse `Airliner` *die Schnittstelle* `CanGlide`*?*
La classe `Airliner` implémente-t'elle l'interface `CanGlide`?

True	*False*
⊗	☐

Es ist möglich, die ID von einem `Plane` *zu lesen?*
Il est possible d'obtenir l'ID d'un `Plane` ?

True	*False*
⊗	☐

Es ist möglich eine zweite Schnittstelle - auch mit dem Namen `Glider` *- zu erstellen, nur für* `Rocket`*?*
Il est possible de créer une deuxième interface, également nommée `Glider`, qui serait spécifique aux `Rocket` ?

True	*False*
☐	⊗

Kann eine Methode der Klasse `Rocket` *das Attribut* `ID` *ändern?*
Une méthode de la classe `Rocket` peut-elle modifier l'attribut `ID` ?

True	*False*
☐	⊗

Eine Instanz der Klasse `Cargo` *besitzt ein ID Attribut?*
Une instance de la classe `Cargo` possède-t'elle un attribut

This question is not valid, should not be considered

True	*False*
⊗	☐

Kann die Methode `getID` *in der Klasse* `Airliner` *überladen werden?*
Peut-on redéfinir la méthode `getID` dans la classe `Airliner` ?

True	*False*
⊗	☐

	True	*False*
Muss man die Methode `getID` *in der Klasse* `Plane` *implementieren?* Doit-on écrire le code de la méthode `getID` dans la classe `Plane`?	☐	⊗
Implementiert die Klasse `Plane` *die Schnittstelle* `HasPassengers`*?* La classe `Plane` implémente-t'elle l'interface `HasPassengers` ?	☐	⊗
Kann eine Methode der Klasse `Cargo` *das Attribut* `maxSpeed` *ändern?* Une méthode de la classe `Cargo` peut-elle modifier l'attribut `maxSpeed` ?	⊗	☐

Fin|Ende