Venga – Project Documentation

0.1

1 Behavioural requirements

- Add training cycle
- Add user
- Check training
- Edit training plan
- Edit training cycle
- Edit user
- Modify training
- Remove training plan
- Remove training cycle
- Remove user
- Set training plan
- Set user

2 Use cases

Set training

The user provides the system with the training start date and duration, and selects the training cycle. The system records the training details and displays the details for confirmation.

Check training

The system displays the current user and the training for the current week (and for the current day). If the user does not have any training set or the user's training has finished, an information is displayed.

Modify training

The user changes one or more of the following: the remaining duration of the training, the point in the training cycle or the entire training cycle. The system displays the details for confirmation.

Add user

The user provides the system with the user's name, the system records the name, displays a confirmation and records the user as selected.

Set user

The user selects his/her name from a list, the system records the selected user and displays the training for the current week.

3 UML Diagrams

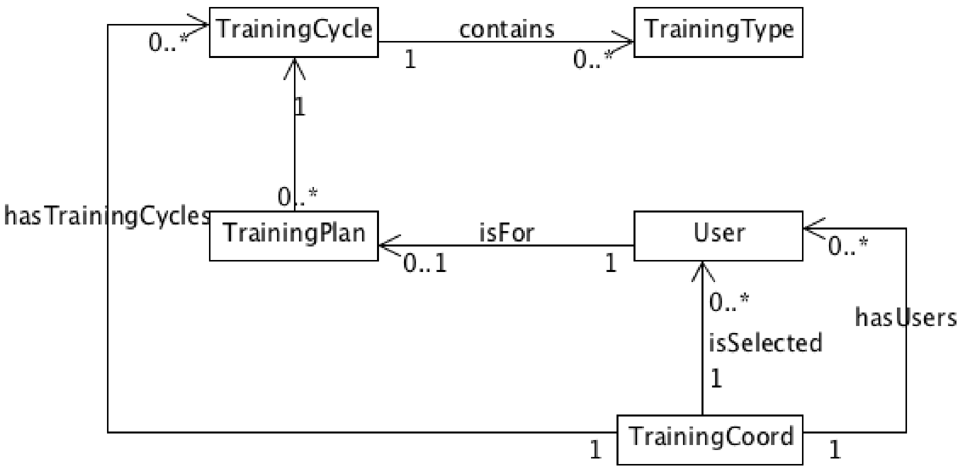

Figure 1   Class diagram

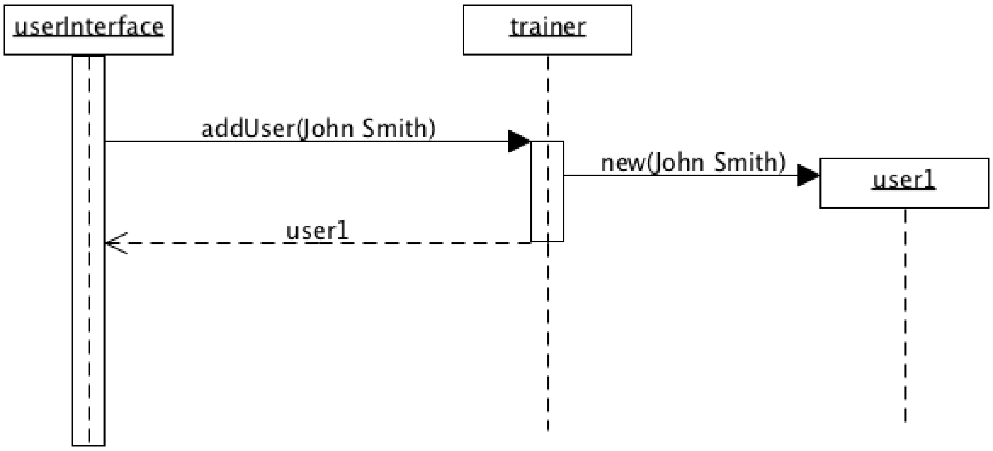

Figure 2 addUser - sequence diagram

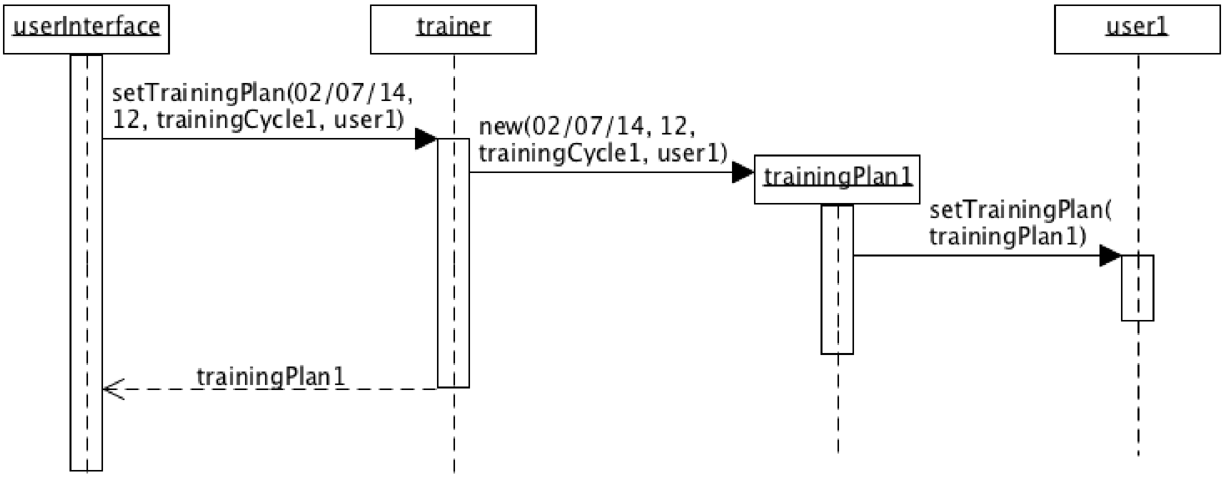

Figure 3 setTrainingPlan - sequence diagram

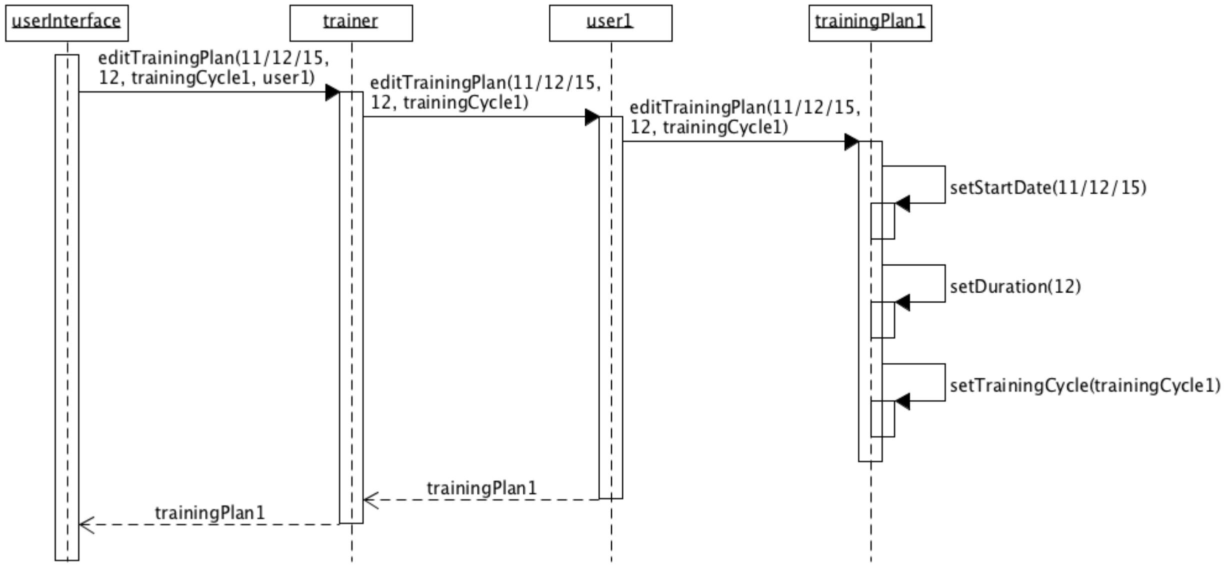
userInterface
trainer
user1
trainingPlan1
editTrainingPlan(11/12/15, 12, trainingCycle1, user1)
editTrainingPlan(11/12/15, 12, trainingCycle1)
editTrainingPlan(11/12/15, 12, trainingCycle1)
setStartDate(11/12/15)
setDuration(12)
setTrainingCycle(trainingCycle1)
trainingPlan1
trainingPlan1

Figure 4 editTrainingPlan - sequence diagram

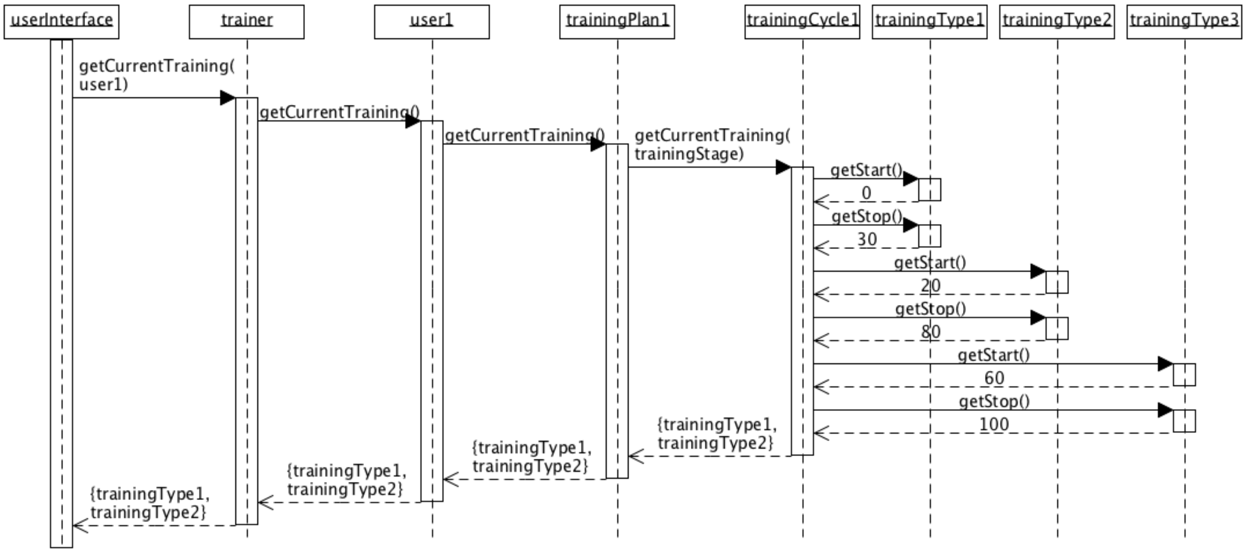

Figure 5 getCurrentTraining - sequence diagram

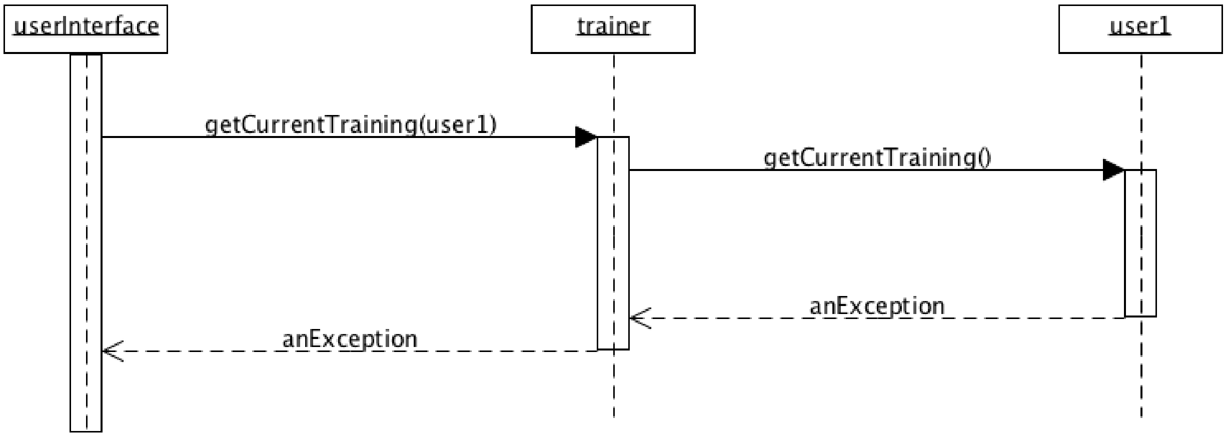

Figure 6   getCurrentTraining - alternative scenario - sequence diagram

4 Implementation model

Class `TrainingCoord`   The coordinating class

Attributes

None

Links

`private User user`

References the linked `User` object.

`private ObservableList <User> users`

References an observabel list of all `User` objects.

`private ObservableList <TrainingCycle> trainingCycles`

References an observable list of all `TrainingCycle` objects.

`private static TrainingCoord trainer`

References the coordinating object if one exist.

Protocol

```
public User addUser(Name aName)
```

Post-condition: a new `User` object, `aUser`, is created with `name` set to `aName`, and:

(i) `aUser` is linked to the `TrainingCoord` object via `hasUsers`;

(ii) `aUser` is returned.

```
public TrainingPlan editTrainingPlan(LocalDate aStartDate, int
aDuration, TrainingCycle aTrainingCycle, User aUser)
```

Pre-conditions: there is a `TrainingPlan` object linked to `aUser`.

Post-condition: in the `TrainingPlan` object linked to `aUser`:

(i) `startDate` is set to `aStartDate`;

(ii) `duration` is set to `aDuration`;

The `TrainingPlan` is linked to `aTrainingCycle` and is returned.

```
public User editUser(User aUser, Name aName)
```

Post-coditions: `aUser's` `name` is set to `aName`.

```
public Collection <TrainingType> getCurrentTraining(User aUser)
```

Pre-conditions: there is a `TrainingPlan` object linked to `aUser`.

Post-conditions: collection of all `TrainingType` objects, linked to the `TrainingCycle` object, linked to the `TrainingPlan` object, linked to `aUser` object, with `start` less than or equal to, and `stop` greater than or equal to, the percentage value of `TrainingPlan` object's `currentWeek` attribute to its `duration` attribute.

```
public User getUser()
```

Post-coditions: returns the linked `User` object.

```
public Collection <User> getUsers()
```

Post-conditions: returns a collection of all the `User` objects.

```
public static TrainingCoord getTrainer()
```

Post-coditions: Creates and returns a new `TrainingCoord` object. Reads in the state of the object from the file Training.data; if there is no such file, or if it is not compatible, returns the object in its initial state.

```
public Collection <TrainingCycle> getTrainingCycles()
```

Post-conditions: returns a collection of all the `TrainingCycles` objects.

```
public TrainingPlan getTrainingPlan(User aUser)
```

Post-coditions: Returns the TrainingPlan object linked to aUser of null if there is no TrainingPlan linked.

```
public void removeTrainer()
```

Post-coditions: destroys a TrainingCoord object if one exists.

Required for testing purposes.

```
public void removeTrainingPlan(User aUser)
```

Pre-conditions: there is a `TrainingPlan` object linked to `aUser`.

Post-coditions: the link from `aUser` to its linked `TrainingPlan` object is removed.

```
removeUser(User aUser)
```

Post-coditions: all links to `aUser` are destroyed; if the receiver was linked to `aUser` via `isSelected` and if it is linked to another `User` object via `hasUsers`, the receiver is now linked to that `User` object via `isSelected`.

```
public void save()
```

Post-conditions: Saves the state of the receiver to the file Training.xml.

```
public TrainingPlan setTrainingPlan(LocalDate aStartDate, int
aDuration, TrainingCycle aTrainingCycle, User aUser)
```

Post-condition: a new `TrainingPlan` object, `aTrainingPlan`, is created with the supplied attribute values, and:

(i) `aTrainingPlan` is linked to `aUser` object;

(ii) aTrainingPlan is linked to aTrainingCycle;

(iii) aTrainingPlan is returned.

```
public void setUser(User aUser)
```

Post-coditions: aUser is linked to the TrainingCoord object via hasUsers.

```
public String toString()
```

Post-condition: returns a string representation of the receiver's user, users and trainingCycles.

Class TrainingPlan The training plan in the system

Attributes

private LocalDate startDate	The start date of the training (should be Monday)
private int duration	The duration of the training (in weeks)
[int /currentWeek	The number of the current week]

Links

```
private TrainingCycle trainingCycle
```

Reference the linked TrainingCycle object.

Constructor

```
TrainingPlan(LocalDate aStartDate, int aDuration, User aUser,
TrainingCycle aTrainingCycle)
```

Post-conditions: initialises a new TrainingPlan object, aTrainingPlan, with the given attribute values. aUser records a reference to the receiver. aTrainingPlan records a reference to aTrainingCycle.

Protocol

```
void editTrainingPlan(LocalDate aStartDate, int aDuration,
TrainingCycle aTrainingCycle)
```

Post-coditions: Sets the attributes of the receiver to the supplied values, the receiver records a reference to `aTrainingCycle`.

`Collection <TrainingType> getCurrentTraining()`

Post-conditions:

(i) calculates `trainingStage`, the percentage value of `currentWeek` to `duration`,

(ii)returns a collection of all the `TrainingType` objects linked to the linked `TrainingCycle` object, with `start` less than or equal to, and `stop` greater than or equal to, `trainingStage`.

`public int getCurrentWeek()`

Post-conditions: if the `startDate` is after the current date or the difference between the `startDate` and the current date is greater than `duration` return `0`, otherwise returns `currentWeek`.

`public int getDuration()`

Post-conditions: returns `duration`.

`public LocalDate getStartDate()`

Post-conditions: returns `startDate`.

`public TrainingCycle getTrainingCycle()`

Post-conditions: returns `trainingCycle`.

`private void setDuration(int aDuration)`

Post-coditions: sets `duration` to `aDuration`.

`private void setStartDate(LocalDate aStartDate)`

Post-coditions: sets `startDate` to `aStartDate`.

`private void setTrainingCycle(TrainingCycle aTrainingCycle)`

Post-coditions: The receiver is linked to `aTrainingCycle`.

`public String toString()`

Post-condition: returns a string representation of the receiver's `startDate` and `duration`.

Class `TrainingCycle` The training cycle in the system

Attributes

`private String name` The unique name of the training cycle

Links

`private Collection <TrainingType> trainingTypes`

References a collection of all the linked `TrainingType` objects

Protocol

`Collection <TrainingType> getCurrentTraining(int aTrainingStage)`

Post-conditions: returns a collection of all the linked `TrainingType` objects, with `start` less than or equal to, and `stop` greater than or equal to `aTrainingStage`.

`public String getName()`

Post-conditions: returns `name`.

`public String toString()`

Post-condition: returns a string representation of the receiver's `name`.

Class `TrainingType` The type of training in the training cycle

Attributes

`private String name`	The name of the training type
`private Colour colour`	The colour of the training type in the system
`private int start`	The start point of the training type in the training cycle expressed as percentage (0 - 100)
`private int stop`	The stop point of the training type in the training cycle expressed as percentage (0 -100)
`private int numOfSession`	The number of training sessions per week

Links

None

Protocol

`public Colour getColour()`

Post-conditions: returns `colour`.

`public String getName()`

Post-conditions: returns `name`.

```
public int getNumOfSessions()
```

Post-conditions: returns `numOfSessions`.

```
public int getStart()
```

Post-conditions: returns `start`.

```
public int getStop()
```

Post-conditions: returns `stop`.

```
public String toString()
```

Post-condition: returns a string representation of the receiver's `name`, `colour`, `start` and `stop`.

Class `User` The user of the system

Attributes

`private Name name` The name of the user

Links

```
private TrainingPlan trainingPlan
```

References the linked `TrainingPlan` object.

Constructor

```
User(aName)
```

Post-conditions: initialises a new `User` object, `aUser`, with `name` set to `aName`.

Protocol

```
TrainingPlan editTrainingPlan(LocalDate aStartDate, int aDuration,
TrainingCycle aTrainingCycle)
```

Pre-conditions: there is a `TrainingPlan` object linked to `aUser`.

Post-coditions: Sets the attributes of the linked `TrainingPlan` object to the supplied values; the linked `TrainingPlan` object records a reference to `aTrainingCycle`; the linked TrainingPlan object is returned.

```
Collection <TrainingType> getCurrentTraining()
```

Pre-conditions: there is a linked `TrainingPlan` object.

Post-conditions: returns a collection of all `TrainingType` objects, linked to the `TrainingCycle` object, linked to the linked `TrainingPlan` object, with `start` less than or equal to, and `stop` greater than or equal to, the percentage value of `TrainingPlan` object's `currentWeek` attribute to its `duration` attribute.

`public Name getName()`

Post-conditions: returns `name`.

`TrainingPlan getTrainingPlan()`

Post-coditions: returns the `TrainingPlan` object linked to the receiver or `null` if there is no `TrainingPlan` linked.

`void removeTrainingPlan()`

Pre-conditions: there is a `TrainingPlan` object linked to the receiver.

Post-coditions: the link from the receiver to its linked `TrainingPlan` object is removed.

`void setTrainingPlan(TrainingPlan aTrainingPlan)`

Post-condition: a reference to `aTrainingPlan` is recorded.

`public String toString()`

Post-condition: returns a string representation of the receiver's `name`.

Class `Name` A person's name

Attributes

`private String title`	The title of the person
`private String firstName`	The first name of the person
`private String surname`	The surname of the person

Constructor

`public Name(String aTitle, String aFirstName, String aSurname)`

Post-condition: initialises a new `Name` object with the given attribute values.

Protocol

`public String getTitle()`

Post-condition: returns `title`.

`public String getFirstName()`

Post-condition: returns `firstName`.

`public String getSurname()`

Post-condition: returns `surname`.

`public boolean equals(object o)`

Post-condition: returns `true` if `o` is a `Name` object with `title`, `firstName` and `surname` equal to those of the receiver; otherwise returns `false`.

`public int hashCode()`

Post-condition: returns the hashcode of the receiver. Consistent with equals().

`public int compareTo(name aName)`

Post-condition: returns a negative integer if the receiver is alphabetically before `aName`, a positive integer if the receiver is alphabetically after `aName` and zero otherwise. `Name` objects are compared using `surname`, `firstName` and `title` in that order.

`public String toString()`

Post-condition: returns a string representation of the receiver's `title`, `firstName` and `surname`.

Invariants

1 For each `TrainingPlan` object, the value of its `currentWeek` attribute is equal to the difference (in weeks) between the current date and the value of its `startDate` attribute.

5 Test cases

Class `TrainingCoord`	
Method `addUser(aName)`	
test 1	Test for adding a new user to the system without any users.
test 2	Test for adding a new user to the system with other user already added to the system.
Method `editUser(aUser, aName)`	
test 1	Test for changing the name of the user.

Method `editTrainingPlan(aStartDate, aDuration, aTrainingCycle, aUser)`	
test 1	Test for editing a training plan for a user with not training plan set.
test 2	Test for editing a training plan for a user with a training plan set.
Method `getCurrentTraining(aUser)`	
test 1	Test for getting the current training for a user with no training plan set.
test 2	Test for getting the current training from the system with a user selected.
Method `getTrainer()`	
test 1	Test for obtaining an instance of `TrainingCoord` when no instance currently exists and there is no Training.data file.
test 2	Test for obtaining an instance of `TrainingCoord` when no instance currently exists but data to create one can be read from Training.xml file.
test 3	Test for obtaining an instance of `TrainingCoord` when one already exists.
Method `getTrainingPlan(aUser)`	
test 1	Test for getting a training plan for a user with no training plan set.
test 2	Test for getting a training plan for a user with a training plan set.
Method `removeTrainingPlan(aUser)`	
test 1	Test for removing a training plan for a user with not training plan set.
test 2	Test for removing a training plan for a user with a training plan set.
Method `removeUser(aUser)`	
test 1	Test for removing a user that is not selected.
test 2	Test for removing a user that is selected and there is another user in the system.
test 3	Test for removing a user that is selected and there is no other in the system.
Method `save()`	
test 1	Test for saving the coordinating object.
Method `setTrainingPlan(aStartDate, aDuration, aTrainingCycle, aUser)`	
test 1	Test for setting a training plan with the start date after the current date, for a user with no training plan set.
test 2	Test for setting a training plan with the start date before the current date, for a user with no training plan set.
test 3	Test for setting a training plan with the start date before the current date, for a user with another training plan set.
Method `setUser(aUser)`	
test 1	Test for selecting the current user in the system with no user currently selected
test 2	Test for selecting the current user in the system with another user currently selected

Table 1 Test cases for the `TrainingCoord` class

Class	`TrainingCycle`
Method	`getCurrentTraining(aTrainingStage)`
test 1	Test for requesting the current training for given training stages.

Table 2 Test cases for the `TrainingCycle` class

Class	`TrainingPlan`
Method	`editTrainingPlan(aStartDate, aDuration, aTrainingCycle)`
test 1	Test for edititing an existing training plan.
Method	`getCurrentWeek()`
test 1	Test for requesting the current week in the training plan that started in the current week.
test 2	Test for requesting the current week in the training plan that will start next week.
test 3	Test for requesting the current week in the training plan has already finished.
Method	`getCurrentTraining()`
test 1	Test for requesting the current training from the training plan that will start the next week.
test 2	Test for requesting the current training from the training plan that is currently in its first week.
test 3	Test for requesting the current training from the training plan that is currently in its second week.
test 4	Test for requesting the current training from the training plan that is currently in its third week.
test 5	Test for requesting the current training from the training plan that is currently in its fourth week.
test 6	Test for requesting the current training from the training plan that has finished

Table 3 Test cases for the `TrainingPlan` class

Class	`User`
Method	`editTrainingPlan(aStartDate, aDuration, aTrainingCycle)`
test 1	Test for editing a training plan for a user with not training plan set.
test 2	Test for editing a training plan for a user with a training plan set.
Method	`getCurrentTraining()`
test 1	Test getting the current training for a user that does not have any training set.
test 2	Test getting the current training for a user that has a training set.
Method	`getTrainingPlan()`
test 1	Test for getting a training plan for a user with no training plan set.

test 2	Test for getting a training plan for a user with a training plan set.
Method `removeTrainingPlan()`	
test 1	Test for removing a training plan for a user with not training plan set.
test 2	Test for removing a training plan for a user with a training plan set.
Method `setTrainingPlan(aTrainingPlan)`	
test 1	Test for setting a new training plan for a user with no training plan set.
test 2	Test for setting a new training plan for a user with another training plan already set.

Table 4 Test cases for the `User` class