Q (https://profile.intra.42.fr/searches)
adaifi

SCALE FOR PROJECT OCAML - OBJECT ORIENTED PROGRAMMING - 1 (/PROJECTS/ OCAML-OBJECT-ORIENTED-PROGRAMMING-1)

You should evaluate 1 student in this team

★

Git repository

```
git@vogsphere-v2.1337.ma:vogsphere/intra-uuid-d79bd967-09af-4aaa
```

Introduction

For the good of this evaluation, we ask you to:

- Stay mannerly, polite, respectful and constructive dunring this evaluation. The trust between you and the 42 community depends on it.

- Bring out to the graded student (or team) any mistake she or he might did.

- Accept that there might be differences of interpretation of the subject or the rules between you and the graded student (or team). Stay open minded and grade as honnestly as possible.

Guidelines

- You must grade only what is present and the graded student's (or team) repository.

- You must stop grading at the first failed exercice, but you are encouraged to continue testing and discussing the following exercices.

Attachments

⬜ subject.pdf (https://cdn.intra.42.fr/pdf/pdf/144423/en.subject.pdf)

Preliminaries

This section is dedicated to setup the evaluation and to test the prerequisits. It doesn't rewards points, but if something is wrong at this step or at any point of the evaluation, the grade is 0, and an approriate flag might be checked if needed.

Respect of the rules

- The graded student (or team) work is present on her or his repository.

- The graded student (or team) is able to explain her or his work at any time of the evaluation.

- The general rules and the possible day-specific rules are respected at any time of the evaluation.

$\checkmark$ Yes $\times$ No

OCaml Object Oriented Programming - 1

- For each exercice, you must compile the exercice using ocamlopt and run the generated executable. If the compilation fails or warns, or an exception is thrown at runtime, the exercice is failed. - Remember to check function names, types, behaviours and outputs.

Ex00, Do What I do. Hold tight and pretend it's a plan!

Test the people class. It should contain :

- a name attribute of type string
- an hp attribute of type int initialized to 100.
- a to_String method that returns a read-friendly string.
- a talk method that should write "I'm [NAME]! Do you know the Doctor?" on the standard output.
- a die method that writes "Aaaarghh!" on the standard output.
- an initializer that prints at least the name and the type of the object. (i.e. "Class people instanciated as Amelia_Pond").

ALL tests MUST be implemented by the student. Again, if something is missing, the feature won't be graded.

$\checkmark$ Yes $\times$ No

Ex01, The Name Of The Doctor!

Test the doctor class. It should contain :

- a name attribute of type string
- an age attribute of type int
- a sidekick attribute that is an object of the people class.
- an hp attribute of type int initialized to 100.
- a to_String method that returns a read-friendly string.
- a talk method that should write "Hi! I'm the Doctor!" on the standard output.

- an initializer that prints at least the name and the type of the object. (i.e. "Class doctor instanciated as The_Tenth").
- a travel_in_time method that do nothing to doctor age (Duh!) and draw a TARDIS on the standard output.
- an use_sonic_screwdriver method that prints on the standard output "Whiiiiwhiiiwhiii Whiiiiwhiiiwhiii Whiiiiwhiiiwhiii".
- a private method regenerate that set the doctor hp to 100.

ALL tests MUST be implemented by the student. Again, if
something is missing, the feature won't be graded.

    ☑ Yes   ✕ No

Ex02, You are a good Daaaaaalek!

Test the dalek class. It should contain :

- a name attribute of type string, generated randomly according to the pattern DalekXXX so DalekSec DalekSup are good. DalekTaMere is wrong.
- an hp attribute of type int initialized to 100.
- a shield attribute of type boolean, mutable, initialized to true and change it's value each time the exterminate method is used.
- a to_String method that returns a read-friendly string.
- a talk method that should write on of the 4 sentences on the standard output.
- an exterminate method that takes an instance of type people as parameter and kills it instantly.
- a die method that prints "Emergency Temporal Shift!" on the standard output.
- a private method regenerate that set the doctor hp to 100.

ALL tests MUST be implemented by the student. Again, if
something is missing, the feature won't be graded.

    ☑ Yes   ✕ No

Ex03, The Day of The Doctor!

Check the army parameterized class :

- if the class is not a parameterized class... Well sorry for you but it's a big fail.
- the class should contains an attribute of type 'a list composed of objects from one of the previsous classes (doctor, people or dalek.)
- It should have an add method allowing to add an object to the list (front or back are ok).
- It should have a delete method that removes an object from the list (front or back are ok).

    ☑ Yes   ✕ No

Ex04, The Time War

Check the gallifrey class. It should have :

- a dalek list attribute.
- a doctor list attribute.

- a people list attribute.
- a do_time_war method that initiate the most epic battle of all space and time.
- any other method or enemy types to improve the Time War dramatically if you find them usefull.
  Remember the student must EXPLAIN any choice he made.

$\oslash$ Yes $\times$ No

Ratings

Don't forget to check the flag corresponding to the defense

✔ Ok

Empty work	📇 Incomplete work	💀 Invalid compilation	🗐 Cheat	♟ Crash

👥 Incomplete group	⚠ Concerning situation	⊘ Forbidden function	💬 Can't support / explain code

Conclusion

Leave a comment on this evaluation

Finish evaluation