Remember that the quality of the defenses, hence the quality of the of the school on the labor market depends on you. The remote defences during the Covid crisis allows more flexibility so you can progress into your curriculum, but also brings more risks of cheat, injustice, laziness, that will harm everyone's skills development. We do count on your maturity and wisdom during these remote defenses for the benefits of the entire community.

# SCALE FOR PROJECT COMPUTORV2 (/PROJECTS/42CURSUS-COMPUTORV2)

You should evaluate 1 student in this team

★

Git repository

```
git@vogsphere.msk.21-school.ru:vogsphere/intra-uuid-c93e
```

## Introduction

For the smooth running of this evaluation, please respect the following rules:

- Remain polite, kind, respectful and constructive whatever happens during this conversation. It's a matter of confidence between you and the 42 community.

- Highlight the potential problems you ‘ve had with the work you're presented to the person or the group you're grading, and take the time to talk about and discuss those issues.

- Accept the fact that the exam subject or required functions might lead to different interpretations. Listen to your discussion partner's perspective with an open mind (are they right or wrong ?) and grade them as fairly as possible.
42's teaching methods can make sense only if peer-evaluation is taken seriously.

## Guidelines

- You must only evaluate what you will find in the student's or group's GiT repository.

- Take the time to check that the GiT repository matches the student or group and the project.

- Double check that no malicious alias was used to mislead you and make you grade something different from the official repository content.

- If a script supposed to help evaluate the exam is supplied by either side, the other side will have to strictly check it to avoid nasty surprises.

- If the evaluating student has not yet taken this project, they will have to read the exam subject in its entirety before starting the evaluation.

- Use the flags available on this grading system to signal an empty or non-funcional project, a norm flaw, cheating, etc. In that case, evaluation stops and final grade is 0 (or -42 if it's a cheating problem). However, if it's not a cheating problem, you are invited to keep talking about the work that has been done (or not done, as a matter of fact) in order to identify the issues that lead to this stalemate and avoid it next time.

- Check the code to make sure no library that would ease calculation and parsing has been used.

# Attachments

subject.pdf (https://cdn.intra.42.fr/pdf/pdf/13228/en.subject.pdf)

# Preliminary part

*In this part, you will just have to check that the evaluated student doesn't use prohibited tools that would ease the acheivement of the project, whether it is a complex type in the chosen language or anything else.*

### Preleminaries

Once you have cloned the repo, ask the evaluated student to set up the work environment to show their work. You can cease the opportunity to check a code has been created to manage the various variables such as:

- Natural whole numbers.
- Rational numbers.
- Complex numbers (with rational coefficients)
- Matrixes
- Second or lower degree polynomial equations.

You should also check that the program compiles and/or executes correctly. During this evaluation, the program should NEVER quit by surprise (Segfault, interpretation error...). If the project doesn't respect one of these key points, the grade is 0 and the evaluation stops.

✓ Yes ✕ No

### Basic checks

Ask the evaluated student to explain how they manage the parsing and the different types (complex, matrixes). If they use a library to help them, the grade is 0 and evaluation stops.

✓ Yes ✕ No

# Assignation part

*In this part, you will test all the behaviours tied to the assignation of a variable or a function. Long tests are suggested. Evaluation will give you test ideas.*

### Basic error test

You will test common errors such as x == 2, or keyboard rollfaces and nonsenses such as, x = 23edd23-+-+

✓ Yes ✕ No

### Semi-advanced error test

This one will be more vicious. Try things like = 2, 3 = 4 or x = g when g has not been defined! Try biased syntaxes such as f(x = 2, ou x = [[4,2].

✓ Yes ✕ No

### Advanced error test

Try the wackiest occurences you can think of. For instance, x = --2, f(x) = x * 2 then t = f(x) (which is impossible), i = 2 (knowing the user cannot assign the variable i). Run anything that might come to your mind.

✓ Yes ✕ No

### Basic valid test

For the following tests, use this "variablename = ?" to find out the value that has been attributed to the variable in the program's context. For instance, if you type x = 2, you can type x = ? and you're supposed to see 2 appear on the next line of the program's interface. In this part, you will test operations such as x = 2, y = 4i, z = [[2,3];[3,5]].

✓ Yes ✕ No

### Semi-advanced valid test

This is where you'll be testing the function and inter-variable assignation.
Don't be afraid to play with spaces and tabs that have to be managed. Test
x = 2 then y =x, and y = ?. You should also test x = 2 then x=5 and x = ?,
If x is not 5, the grade is 0.
You can try the same with matrixes or imaginary numbers, such as
A = [[2,3]], then B = A. If B = ? doesn't show A, the grade is 0.

✓ Yes ✕ No

### Advanced valid test

For this question, you will test assignations with many elements. Start with
x = 2, y = x * [[4,2]], f(z) = z * y. If f(z) = ? doesn't show z * [[8,4]],
student gets 0.
You will also test x = 2 then f(x) = x * 5. If f(x) = ? doesn't show 10
(or something similar, like 2 * 5 for instance), student also gets 0.
Don't be afraid to test anything you might think of, with any kind of types,
imaginary matrixes and so on as long as it makes mathematically sense.

✓ Yes ✕ No

# Calculation part

*In this part, you will test behaviors tied to calculations as well as a function evaluation. Long testing is suggested. Test ideas will appear during the evaluation.*

### Basic valid test

This is where you test very simple calculations such as 2 + 2 = ?,
3 * 4 = ?, x = 2 and then x + 2 = ?. You can also try to divde by 0, like
2 / 0 = ?. You will also try the float management like in 1.5 + 1 = ?

✓ Yes ✕ No

### Semi-advanced valid test

Here, you will run little more complex calculations, like x = 2 * i and
x ^ 2 = ?. If the result is not -4, student gets 0.
You should also test matrix multiplication such as A = [[2,3];[3,4]] and
B = [[1,0];[0,1]] then A ** B = ?. You should see the matrix A displayed.
Otherwise ? Student gets a 0. You should also test such inputs as
f(x) = x + 2, p = 4, f(p) = ? If the result is not 6, guess what ? It's a
0!

✓ Yes ✕ No

### Advanced valid test

Run complex calculations such as 4 - 3 - ( 2 * 3 ) ^ 2 * ( 2 - 4 ) + 4 = ?
for instance. Or f(x) = 2*(x + 3*(x - 4)) then p = 2, and
f(3) - f(p) + 2= ? The result is supposed to be 10. You can mix complex
numbers with functions like f(x) = 2 *x * i and f(2) = ? (the expected
result is 4i). Just like the matrixes, try anything. Use the program as if
it were a calculator.

✓ Yes ✕ No

# Bonus

*Reminder: if the program shows some kind of flaws or erratic behavior (bus error, segfault, etc...), the evaluation stops. Students gets a 0. Use the respective flags. This instuction goes for the whole evaluation. Bonus will be taken into account only if the mandatory part is PERFECT. PERFECT meaning it is completed, that its behavior cannot be faulted, even because of the slightest mistake, improper use, etc... Practically, it means that if the mandatory part is not validated, none of the bonus will be taken in consideration.*

### Bonus

Let the evaluation guide you through the implemented bonus. You will be
free to grade them as you see fit.

**Rate it from 0 (failed) through 5 (excellent)**

○

# Ratings

**Don’t forget to check the flag corresponding to the defense**

✔ Ok ★ Outstanding project

Empty work No author file Invalid compilation Norme Cheat

Crash Incomplete group ⊘ Forbidden function

# Conclusion

**Leave a comment on this evaluation**

**Finish evaluation**

Privacy policy (https://signin.intra.42.fr/legal/terms/5)
Legal notices (https://signin.intra.42.fr/legal/terms/3)
Declaration on the use of cookies (https://signin.intra.42.fr/legal/terms/2)
Rules of procedure (https://signin.intra.42.fr/legal/terms/4)
Terms of use for video surveillance (https://signin.intra.42.fr/legal/terms/1)
General term of use of the site (https://signin.intra.42.fr/legal/terms/6)