# SCALE FOR PROJECT OCAML - PATTERN MATCHING AND DATA TYPES - 0 (/PROJECTS/OCAML-PATTERN-MATCHING-AND-DATA-TYPES-0)

You should evaluate 1 student in this team

★

Git repository

```
git@vogsphere-v2.1337.ma:vogsphere/intra-uuid-2fef5b8f-9b7b-49ba
```

## 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/144531/en.subject.pdf)

# Preliminaries

**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.
- For this project, you need to clone the Git repository on the evaluated person's computer.

Yes No

# OCAML - Pattern Matching and Data Types - 0

*- 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. - Whether the graded student provided tests or not, you must test her or his work extensivelly and asses if the work is done or not. - Remember to check function names, TYPES, behaviours and outputs. - Never test overflows for today. - One more time, you HAVE to check the type of the functions since it's essential today*

**Ex00, Do you even compress bro?**

Test the program with at least the following lists: encode ["a";"a";"a";"b";"b";"b"] encode [] encode [(3, 2); (3, 2); (4, 3)]

Yes No

**Ex01, Crossover**

Test the program with at least the following lists : crossover [1;2;3] [1;2;3] crossover ["toto";"tata";"titi"] ["toto";"tata";"tutu"] crossover ["toto";"tata";"titi"] [] crossover [] ["toto";"tata";"titi"]

Yes No

**Ex02, Fifty Strings of Gray**

Test the Gray sequences generation program with at least the following values : 0, 1, 2, 3, 4 and 5. Also if the student has

used the list concatenation operator ( @ ), It's considered as cheating so -42.

Yes No

**Ex03, One and one and one is three**

Test the function with the following values. sequence 1 = "1" sequence -1 = "" sequence 6 = "312211" sequence 8 = "1113213211"

Yes No

**Ex04, DNA -> Nucleotides**

Check the following facts :

- phosphate and deoxyribose types are both string type aliases.
- nucleobase type is a variant composed of A, T, C, G, and None.
- nucleotide type is a record type or a triplet composed of one of every types above (e.g. 1 phosphate, 1 deoxyribose and 1 nucleobase).

Also check the generate_nucleotide function. It should return a nucleotide type from a char as a parameter.

Yes No

**Ex05, DNA -> Helix**

- Check the function generate_helix (1pts)
- Check the function helix_to_string (1pts)
- Check the function complemetary_helix (for example ATCG should output TAGC) (3pts)

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

**Ex06, DNA -> Messenger RNA**

Check the function generate_rna. The ATCG helix must produce an UAGC rna.

Yes No

**Ex07, DNA -> Ribosome**

Check the function generate_base_triplets. it should return a nucleotide triplet list from an rna as parameter. If the number of nucleotides in the rna is not a factor of 3, the triplet should be forced filled with None elements (Or anything that can represent the Great Interstellar Void).

Check the decode_arn function. It should return a protein (a list of aminoacids) from an element of type rna as parameter. Use a couple different arn sequences of your choice to test this using the string_of_protein function.

Yes No

**Ex08, DNA -> The Complete Process of Protein Creation**

This one is a gift, check that the function is functionning as intented. Life is amazing isn't it ? For example the parameter "TACTACATCTAC" should output something like Met-Met-STOP in the end. Also check that each step is explained toroughly and clearly outputed. Remember that the decoding function MUST stop the process at the first STOP occurence.

Yes 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**

legal/terms/2) legal/terms/6) legal/terms/4) profile.intra.42.fr/legal/terms/1)