VE477
Introduction to Algorithms

Lab 1

Manuel — UM-JI (Fall 2021)

Goals of the lab

- Review on dictionaries
- Review on hash tables
- Starting with OCaml

1 C programming

Using the C standard library, write an efficient implementation of the following data structures:

- Dictionary with the seven basic operation. Use sorted double link lists.
- Implement hash tables.

2 Functional programming

Starting with lab 3 you will be highly encouraged to use OCaml in order to complete the tasks (up to +2.5 marks on the final grade). The purpose of the two first labs is to help you getting acquainted with the functional programming paradigm and OCaml language.

Although historically less popular, functional programming has started to gain more exposure in the recent years and has been adopted by major actors in the industry. Functional languages often feel misleading and almost "counter-intuitive" to new comers, but are highly praised for simplifying the writing of predictable, reliable, and stable code, hence highly decreasing the possibility of bugs.

Gaining a good understanding in OCaml throughout VE477 labs should be a significant addition to your base knowledge that you could then value in your CS related graduate school or internship applications.

Questions:

- What are imperative and object oriented languages? Give examples.
- Functional programming:
- What does it mean for a function to be a *first class citizen*?
- What is a *higher order function*?
- Using basic mathematics show that integration defines is a linear map. How does it relate to higher order functions?
- What does it mean for a variable to be *immutable*?
- What are the pros and cons of dealing with immutable variables?
- What is a *pure function* or a function with *no side-effects*?

Places of interest:

- Programming paradigms
- OCaml
- Functional programming

3 Getting started with OCaml

OCaml is a general purpose language following multi-programming paradigms, namely *functional*, *imperative*, and *object oriented*. The purpose of this part is to get acquainted with OCaml development environment. So first, start by installing OCaml following the Up and Running tutorial. Note: as for most development environment, installation might be trickier on Windows than on other OSes.

Follow the First Hour with OCaml tutorial and answer the following questions:[1]

- What is the meaning of REPL?
- Is OCaml an interpreted or a compiled language?
- Which is "faster", C or OCaml?
- What is the `let` keyword used for?
- What are the basic types in OCaml?
- What is type inference?
- What is the meaning of `f : int -> int -> float`?
- What is the "`NULL` reference problem"? How does the keyword `option` help in OCaml?
- What is the difference between statically and dynamically typed languages? Which category does OCaml belong to?
- What is pattern matching. Provide a simple OCaml example different from the one in the tutorial.
- How does pattern matching increase type safety?

To conclude, have a brief look at OCaml Programming Guidelines. When writing OCaml code in future labs, do not forget to refer to them for more details.

4 Interview problems

Given a set of integers, design an algorithm which finds all the subsets composed of three elements summing up to 0. Analyze its time and space complexity.

Given a sorted linked list, describe a strategy to delete all nodes featuring duplicate numbers. The result should be a linked list only composed of nodes featuring distinct numbers.

[1] Some question might require checking other resources.