

## Assignment 2A

Due Friday February 27 at 11:50am, just before tutorial.

No late assignments will be accepted.

---

The questions below require you to write ML functions. These are warm-up exercises to familiarize you with ML, and with typed functional programming. Unless stated otherwise, (i) all programs should be purely functional (*i.e.*, have no side effects), (ii) pattern matching should be used whenever possible, and (iii) simple programs are preferred to complex ones. Feel free to use helper functions wherever appropriate. You should hand in two files: a source listing of all your ML functions, and a sample terminal session with the ML interpreter. These files should be submitted electronically using the submission web page. The source code should be well commented, and the terminal session should be short but should demonstrate that your functions work correctly. To keep things simple, you may assume that the input to your functions is correct, so that no error checking is required.

The marker has a limited amount of time for each assignment, so it is your responsibility to provide documentation and testing that allows him to *quickly* evaluate your work. As with all work in this course, 20% of the grade is for quality of presentation.

---

1. (4 points total) Define an ML function (`sumTimes L`) that takes a list `L` of integer pairs, and returns the sum of the product of each pair of integers. For example, (`sumTimes [(1,2), (3,4), (5,6)]`) returns 44 (*i.e.*,  $1*2 + 3*4 + 5*6$ ). Define the function in two ways: (a) without pattern matching (2 points), and (b) with pattern matching (2 points). (These two versions of the function may be given different names, such as `sumTimes1` and `sumTimes2`.)

2. (11 points total)
  - (a) (1 point) Define `employee` to be a named type for records, where each record has four fields: `id`, `name`, `age`, `salary`.
  - (b) (2 points) Define an ML function (`avgAge L`) that takes a list `L` of employee records, and returns the average age of the employees.
  - (c) (2 points) Define an ML function (`payroll L`) that takes a list `L` of employee records, and returns the sum of their salaries.
  - (d) (3 points) Define an ML function (`personnel L`) that takes a list `L` of employee records, and returns a list of pairs, where each pair has the form `(id,name)`, one for each employee record in `L`.
  - (e) (3 points) Define an ML function `increase(L1,L2)`, where `L1` is a list of employee records and `L2` is a list of integers, where `L1` and `L2` have the same length. The function returns a list of employee records identical to `L1` except that each employee's salary has been increased. Specifically, the function adds the  $i^{th}$  integer in `L2` to the salary of the  $i^{th}$  employee in `L1`.
  
3. (7 points total)
  - (a) (1 points) Define an enumerated datatype called `grade` whose values are `A`, `B`, `C`, `D`, and `F`.
  - (b) (2 points) Define a function (`convert N`) that converts a number grade `N` to a letter grade, according to `U` of `T`'s grading scheme. Thus, (`convert 85`) => `A` and (`convert 57`) => `D`.
  - (c) (2 points) Define a function (`gp G`) that converts a letter grade to a grade point number, i.e., to 4, 3, 2, 1 or 0. Thus (`gp A`) => 4, (`gp B`) => 3, etc.
  - (d) (2 points) Define a function (`gpa L`) that takes a list `L` of letter grades, and returns their gpa, i.e., the average of the grade points of the letter grades. Thus, (`gpa [A,B,B]`) => 3.333 (i.e.,  $(4+3+3)/3$ ).
  
4. (11 points total)
  - (a) (2 points) Define a recursive type representing trees where each node can have any number of children, and where each node stores an integer.
  - (b) (3 points) Define a function (`count T`) that returns the number of leaves in tree `T`.
  - (c) (3 points) Define a function (`numbers T`) that returns a list of all the integers in tree `T`.
  - (d) (3 points) Define a function (`double T`) that doubles all the integers in tree `T`. That is, it returns a tree identical to `T` except that the integer stored in each node has been doubled.
  
5. (12 points) Redo Question 4 from Assignment 1b using ML. Define appropriate types to make your code easier to understand. A solution to Q4 in A1b will be posted on the course web page after A1b is handed in. You may simply translate this from Scheme into ML.

## Cover sheet for Assignment 2A

---

Complete this page and attach it to the front of your assignment.

**Name:** \_\_\_\_\_  
(Underline your last name)

**Student number:** \_\_\_\_\_

I declare that this assignment is solely my own work, and is in accordance with the University of Toronto Code of Behavior on Academic Matters.

**Signature:** \_\_\_\_\_