Лабораторная работа №1
«Люди — животные»

Предварительные сведения

В ходе выполнения данной работы необходимо написать 20 функций на языке Standard ML. Во всех нижеперечисленных условиях «дата» — представляется как значение типа `int * int * int` — тройка целых положительных чисел, в которой первое целое число есть номер дня в месяце, второе — номер месяца, а третье — номер года. Корректной датой будем считать даты, заданные по григорианскому календарю.

Прежде чем приступать к выполнению заданий, ознакомьтесь с замечаниями, изложенными в пункте 2 данного описания.

1. Задания

В задании 1 будем считать, что год по григорианскому календарю может принимать значения в диапазоне от 1 до 6999 (проверять этот факт в решении и тестах к решению не следует).

1. В юлианском и григорианском календарях в високосном году в феврале добавляется лишний день, отсутствующий в обычном году, — 29 февраля. Високосный год в григорианском календаре — год, номер которого делится на 400 без остатка или, если не кратен 100, то кратен четырем. Опишите функцию `isLeapYear`, получающую в качестве аргумента номер года и возвращающую `true`, если год високосный, или `false` — в противном случае.

   Типовое решение — 3 строки кода.

2. Опишите функцию `isCorrectDate`, получающую в качестве аргумента кортеж из трех целых чисел `(d, m, y)` и возвращающую `true`, если этот кортеж представляет корректную дату, и `false` — в противном случае.

   Прежде чем приступать к решению, стоит вспомнить количество дней в каждом месяце года. При необходимости проверки того, является ли год високосным, необходимо использовать функцию `isLeapYear`.

   Решение — около полутора десятков строк форматированного кода.

В последующих задачах проверять на корректность даты, переданные функциям в качестве аргументов, не следует. Считаем, что функциям передаются только корректные даты. В задачах 4–10 будем считать, что год по григорианскому календарю может принимать значения в диапазоне от 201 до 6999 (проверять этот факт в решении и тестах к решению не следует).

3. В юлианском календаре високосным годом считается год, номер которого просто кратен четырем без дополнительных условий. Поэтому юлианский календарь «отстает» от григорианского, и это отставание увеличивается каждые 100 или 200 лет.

   Опишите функцию `newStyleCorrection`, получающую григорианскую дату `(d, m, y)` в качестве аргумента и возвращающую количество дней от нее до даты `(d, m, y)` по юлианскому календарю (т. е. количество дней «поправки» между юлианским (старый стиль) и григорианским (новый стиль) календарями на заданную дату). Результат должен вычисляться как количество дней «29 февраля» нашей эры (до заданной даты включительно) в юлианском календаре, отсутствующих в григорианском календаре, минус два. Например, для даты «1 марта 2016 года» функция должна возвращать `13`, а для «28 февраля 1700 года» — `10`.

   Объем решения — десяток строк.

4. Опишите функцию `getNthInt`, получающую список целых чисел и целое `n`. Функция должна вернуть `n`-ый элемент заданного списка, при условии, что элементы нумеруются с нуля. Считаем, что количество элементов в заданном списке превышает `n` (проводить проверку этого факта не следует).

   Форматированное решение занимает 4 строки.

5. Опишите функцию `getNthStr`, получающую список строк и целое `n`. Функция должна вернуть `n`-ый элемент заданного списка, при условии, что элементы нумеруются с нуля. Считаем, что количество элементов в заданном списке превышает `n` (проводить проверку этого факта не следует).

6. Опишите функцию `lastSmaller`, получающую целое число `amount` и непустой список положительных целых чисел, упорядоченный по возрастанию. Функция должна выдавать максимальный элемент заданного списка, меньший по величине чем `amount`. Если все элементы данного списка не меньше `amount`, то результатом функции должен быть целочисленный ноль — `0`.

   В типовом решении используется вспомогательная рекурсивная функция, одним из аргументов которой является потенциальный результат вычислений, а вторым — исследуемый список. Благодаря тому, что заданный список упорядочен, функция делает свой вывод, сравнивая голову списка с потенциальным результатом.

   В типовом решении 8 строк

В заданиях 7–12 будем вычислять некоторые значения с точностью до 5-ти знаков после запятой. Но вместо представления чисел в виде вещественных чисел с плавающей точкой будем проводить операции над целыми числами и считать, что `100000` — это представление числа 1.0, а `2735601` — представление числа 27.35601. Будем в дальнейшем здесь называть такое представление представлением с фиксированной точкой.

Для выполнения следующего задания вам потребуются списки чисел `thousandCorrection`, `hundredCorrection`, `decadeCorrection`, `yearCorrection`, `monthCorrection`, `calendarCorrection`, `reductions`, представленные в заготовке решения. Считаем, что пять последних цифр в каждом числе этих списков — знаки после десятичной точки (т. е. числа уже представлены в форме с фиксированной точкой).

7. Опишите функцию `firstNewMoonInt`, получающую дату в качестве аргумента и возвращающую `NONE`, если в месяце заданной даты нет новолуния, и `SOME d`, если `d` — целочисленное значение (`d` $\geqslant 100000$) — номер дня первого новолуния в месяце заданной даты в представлении с фиксированной точкой.

   Так как между двумя новолуниями примерно 29.5 дней, новолуния может не быть только в феврале. В любой другой месяц оно всегда есть.

   Для вычисления результата сначала нужно вычислить количество дней разницы между юлианским и григорианским календарем на заданную дату и представить его в форме с фиксированной точкой. Затем, добавить к результату поправки, содержащиеся на соответствующих позициях в предложенных списках.

   Перед получением поправок нужно уменьшить номер года на единицу, если заданный месяц — январь или февраль. Позиции поправок из первых четырех списков вычисляются как количество тысяч, сотен, десятков и единиц соответственно в заданном номере года, если нумерацию элементов вести с нуля. Позиция поправки из списка `monthCorrection` — номер заданного месяца, если начинать нумерацию элементов с единицы. Позиция в списке `calendarCorrection` — остаток от деления номера года на 4 (при нумерации элементов списка с нуля).

   Из полученной суммы нужно вычесть максимально возможное число из списка `reductions`, значение которого меньше чем сумма без 100000 (без единицы). Если такого числа в списке `reductions` не найдется — оставить сумму без изменения.

   Полученное число — результат функции, который нужно обернуть в контейнер `SOME`, если целая часть числа не больше, чем количество дней в заданном месяце. В противном случае функция должна вернуть `NONE`.

   Если проанализировать содержимое заданных списков, можно выяснить, что полученный результат вычислений не может быть больше чем 3053059 (т. е. 30.53059), что позволяет не делать дополнительные проверки на количество дней в текущем месяце.

   Решение должно использовать описанные ранее функции `newStyleCorrection`, `getNthInt`, `lastSmaller`.

   Объем решения составит порядка двух десятков строк.

8. Опишите функцию `firstNewMoon`, получающую дату в качестве аргумента и возвращающую значение типа `(int * int * int) option`. Результат функции — `SOME d`, если `d` — дата первого новолуния в месяце заданной даты, или `NONE`, если в заданном месяце новолуния нет.

   Функция должна получать результат вызова `firstNewMoonInt` с тем же аргументом, и если результат не `NONE`, отбрасывать от числа в контейнере `SOME` дробную часть (превращая его из числа с фиксированной точкой в целое число без точки) и конструировать результат типа `(int * int * int) option`.

   Типовое решение — 8 строк.

9. Опишите функцию `dateToString`, выдающую для заданной даты строку в виде `"Month Day, Year"`, где `Month` — английское наименование месяца (строчными символами с заглавной буквы), `Day` — номер дня в месяце, `Year` — номер года. Например, `"May 5, 1980"`.

   Для конкатенации используйте бинарную операцию `^`. Для перевода целого числа в строку применяйте функцию `Int.toString`. Для получения наименования нужного месяца используйте список из 12-ти строковых значений и функцию `getNthStr`, описанную ранее. Английские наименования месяцев: January, February, March, April, May, June, July, August, September, October, November, December. В строке результата после номера дня должна стоять запятая. После запятой и перед номером дня должен быть один пробел. Лишних символов в строке быть не должно. Ведущих нулей в номере дня и номере года быть не должно.

   Решение — около 10 строк.

10. Опишите функцию `isOlder`, получающую в качестве аргументов две даты и возвращающую `true`, если первая дата предшествует второй, или `false` — в противном случае.

    Объем решения составит порядка 13 строк.

В заданиях 11–20 будем считать, что год по григорианскому календарю может принимать значения в диапазоне от 201 до 2999 (проверять этот факт в решении не следует).

11. Так как календарный год по продолжительности не совпадает с астрономическим, дата зимнего солнцестояния (самый короткий день в году) каждый год смещается и может приходиться на 20–23 декабря. Каждый обычный год дата зимнего солнцестояния увеличивается на $\delta_1 = 0.2422$ суток, а каждый високосный — уменьшается на $\delta_2 = 0.7578$ суток. Таким образом, можно подсчитать дату, на которую приходится день зимнего солнцестояния, если принять за точку отсчета значение 22.5 — условная дата зимнего солнцестояния в 1-й год до н. э. (год номер 0).

    Напишите функцию `winterSolstice`, получающую номер года в качестве аргумента и выдающую целое число — номер дня в декабре заданного года, на который приходится день зимнего солнцестояния.

    Все операции с дробными значениями следует проводить над числами с фиксированной точкой. Результат функции — целая часть результата вычислений.

    Решение будет проще, если принять во внимание, что $\delta_1 + \delta_2 = 1$.

    Объем форматированного решения — пять строк.

12. Дата нового года по китайскому календарю вычисляется как дата второго новолуния после дня зимнего солнцестояния (даже если на день зимнего солнцестояния выпадает новолуние).

    Опишите функцию `chineseNewYear`, получающую номер года и возвращающую дату китайского нового года в заданном году.

    Считаем, что даты двух соседних новолуний отличаются на `29.53059` суток, а даты новолуний, отстоящих друг от друга через одно, отличаются на `59.06118` суток.

    Функция должна получать результат вызова `firstNewMoonInt` для декабря заданного года, и на основании сравнения полученного номера дня с результатом функции `winterSolstice` отсчитывать нужную дату.

    Все операции с дробными значениями следует проводить над числами с фиксированной точкой.

    Типовое решение — 13 строк.

Для выполнения следующего задания потребуются списки `celestialChi`, `celestialEng`, `celestialColor`, `terrestrialChi`, `terrestrialEng`, представленные в заготовке к решению.

13. Летоисчисление в китайском календаре ведется согласно 60-летнему циклу. Каждому году в цикле дается название из двух частей. Первая часть — один из десяти представителей небесной ветви (небесные стихии). Их китайские и английские наименования представлены в списках `celestialChi` и `celestialEng` соответственно. Каждой паре представителей небесной ветви сопоставляется один из пяти цветов. Наименования возможных цветов приведены в списке `celestialColor`. Вторая часть наименования года — один из двенадцати представителей земной ветви (земные стихии). Их наименования приведены в списках `terrestrialChi` и `terrestrialEng`.

    Для определения года по китайскому календарю нужно к номеру года по григорианскому календарю добавить 2396 и найти остаток от деления полученного числа на 60. Так мы получим порядковый номер года в 60-летнем цикле (начиная нумерацию с нуля). Если найдем остаток от деления полученного номера на 10, то получим порядковый номер небесной стихии. Если найдем целую часть от деления порядкового

номера небесной стихии на 2, получим порядковый номер соответствующего цвета. Если найдем остаток от деления номера года в цикле на 12, то получим порядковый номер соответствующей земной стихии (наименование животного).

Опишите функцию `chineseYear`, получающую в качестве аргумента целое число — номер года по григорианскому календарю, и возвращающую наименование китайского года, начинающегося в заданном году. Результат должен представлять четверку `string * string * string * string`, где первая строка — китайское наименование года — два слова через дефис, а вторая строка — цвет соответствующий небесной стихии, третья строка — наименование животного и четвертая строка строка — английское наименование небесной стихии. Так, например, для года `1980` должен получиться результат `("Geng-Shen", "White", "Monkey", "Metal")`.

Объем форматированного типового решения — 15 строк.

14. Опишите функцию `dateToChineseYear`, получающую дату в качестве аргумента и возвращающую, как и в предыдущим задании, четверку строк наименований для года китайского календаря, соответствующего заданной дате.

    Решение составит порядка 8 строк.

15. Опишите функцию `dateToAnimal`, получающую дату в качестве аргумента и возвращающую строку — английский аналог наименования года по китайскому календарю, состоящий только из наименования животного (то, что в результатах предыдущих двух функций стоит на третьей позиции). То есть, например, для 25 июня 1980 года функция должна выдать `"Monkey"`.

    В решении нужно использовать функцию `dateToChineseYear` в качестве вспомогательной.

    Решение — 2 строки.

16. Опишите функцию `animal`, аргумент которой — пара `string * (int * int * int)`. Первый элемент пары — имя студента (строка), второй — дата его рождения. Функция должна возвращать строку — наименование животного, соответствующее по китайскому календарю дате рождения студента.

    Форматированное типовое решение — 2 строки.

17. Опишите функцию `extractAnimal`, получающую два аргумента: первый — список пар `string * (int * int * int)` — список имен студентов с датами рождения, второй — наименование животного из китайского календаря (строка). Функция должна возвращать список имен студентов с датами их рождения, которым по китайскому календарю соответствует указанное животное.

    Типовое решение — 7 строк.

18. Опишите функцию `extractAnimals`, получающую два аргумента: первый — список имен студентов с датами рождения, второй — список строк — наименований животных из китайского календаря. Функция должна возвращать список имен студентов с датами их рождения для тех студентов, которым по китайскому календарю соответствует какое-либо из животных заданного списка. Порядок следования студентов в списке-результате не имеет значения.

    Предполагаем, что список, передаваемый функции в качестве второго аргумента, не содержит повторяющихся элементов (проверку этого факта проводить не следует).

    Решение — 4 строки.

19. Опишите функцию `oldest`, получающую в качестве аргумента список имен студентов с датами рождения. Функция должна возвращать значение типа `string option`: `NONE`, если заданный список пустой, или `SOME name`, если `name` — имя старшего из студентов в заданном списке. Если в заданном списке два студента имеют один и тот же возраст, то старшим из них будем считать студента, упоминающегося в списке первым.

    Типовое решение — 12 строк.

20. Опишите функцию `oldestFromAnimals`, получающую два аргумента: первый — список имен студентов с датами рождения, второй — список строк — наименований животных из китайского календаря. Функция должна возвращать значение `string option`: `SOME name`, если `name` — имя старшего из студентов, которому по китайскому календарю соответствует какое-либо из животных заданного списка. Если в списке первого аргумента остутствуют «животные» из второго аргумента, то результатом функции должно быть значение `NONE`.

    Форматированное решение — 4 строки.

Общий объем решения, включая строки, предварительно заданные в файле, должен составить 220–250 строк.

2. Замечания по выполнению заданий

2.1. Необходимый минимум

Для выполнения работы потребуются сведения о следующих функциях, операциях и конструкциях:

- конструкции `fun` и `val` для определения функций и переменных
- конструкция `if...then...else...`
- конструкция `let...in...end`
- арифметические операции `+`, `-`, `*`
- целочисленные операции `mod`, `div`
- логические операции `orelse`, `andalso`, `not`
- операции сравнения `<`, `>`, `=`, `<>`, `<=`, `>=`
- конструктор кортежа `(`, `)`
- операция извлечения элемента с номером `n` из кортежа `#n` (`n` — целочисленный литерал)
- конструкторы списка `::` и `[]`
- операции для работы со списками: `hd`, `tl`, `null`, `@`
- функция преобразования целого числа в строку `Int.toString`
- операция конкатенации (сцепления) строк `^`
- конструкторы значений типа `option` — `SOME` и `NONE`
- функции для работы со значениями типа `option`: `valOf` и `isSome`
- конструкция `use` для загрузки функций из файла с заданным именем

Информацию об этих и других конструкциях можно найти в конспекте, приведенном на портале.

2.2. Ограничения

При выполнении данной лабораторной работы нужно соблюдать следующие ограничения:

1. При описании каждой функции должны быть явно прописаны типы аргументов и тип результата;
2. Нельзя использовать функции, не перечисленные в разделе 2.1. Если вы считаете, что для выполнения какого-то из заданий необходима функция/конструкция, отсутствующая в разделе 2.1 — задайте вопрос на форуме «Лабораторная работа №1»;
3. Нельзя использовать механизм сопоставления с образцом (pattern matching).

Можно определять собственные вспомогательные функции. Но вспомогательная функция может быть определена как функция верхнего уровня только если она используется в решениях минимум двух разных заданий. В остальных случаях вспомогательные функции должны быть локальными.

2.3. Предостережения на счет языка

Будьте внимательны:

- строковые литералы заключаются в двойные кавычки, а не в апострофы;
- в каждой конструкции `if` блок `else` является неотъемлемой частью;
- унарный минус обозначается тильдой `~`;
- сравнение на равенство — `=`, а не `==`;
- сравнение на неравенство — `<>`;
- тип указывается после знака `:` после имени переменной;
- вместо привычных `and` и `or` здесь `andalso` и `orelse`.

2.4. Предостережения на счет решения

Решением каждой задачи должна быть функция с указанным именем и возвращающая значение в той форме, в которой спрашивается в задании. Прежде чем отправить решение на проверку проводите сравнение сигнатуры написанной вами функции с соответствующей сигнатурой, приведенной в разделе 2.5.

Тесты, представленные в файле `test-1.sml`, могут помочь, в том числе, и в понимании задания. Для правильно-функционирующего решения представленные тесты (без модификаций) должны выполняться на 100%.

Вычисляемые календарные даты (новолуние, солнцестояние, китайский новый год) могут не совпадать с общепринятыми датами (для XX и XXI веков может быть погрешность до 1 суток). Поэтому, предварительно вычисленные значения, которые Вы можете найти в сети Интернет, можно брать во внимание только как примерный ориентир. Правильность решения определяется как правильность реализации предложенного в задании алгоритма.

Сообщение в задании, например, что год может принимать значения между 201 и 2999 полагает, что когда вы реализуете свои решения, вы должны быть уверены, что на этом промежутке ваша функция должна выдавать правильный результат. Более того, он максимально приближен к реальному ответу (к реальным датам, о которых идёт речь). В тестах к решениям могут встречаться запуски от значений, выходящих за указанный диапазон. В заданиях оговаривается, что вы не должны в своих решениях проверять принадлежность года указанному промежутку. Если ваш алгоритм написан без ошибок, так, как об этом спрашивается в задании, то приведённые тесты сработают верно, хотя полученное решение наверняка не будет совпадать с реальным, так как для дат вне этого промежутка алгоритмы вычислений должны быть другими.

Не следует делать предположений насчет задания, не сформулированных явно в условии. Если возникают сомнения — задайте вопрос на форуме «Лабораторная работа №1».

Присутствие примечания, что в задании не нужно осуществлять проверку какого-то факта, означает, что добавление такой проверки при автотестировании может привести к интерпретации программы как неверно функционирующей.

Избегайте повторений вычислений в одной функции. Вместо того, чтобы вычислять одно и то же значение несколько раз — сохраняйте вычисленное значение в переменной. Исключение можно сделать для результатов функций `hd` и `tl`.

На количество строк решения, указанное в задании, стоит обращать внимание только как на примерный ориентир. Если ваше решение намного меньше по объему, чем указано, то возможно, что Вы учли не все условия, указанные в задании. Если ваше решение намного больше по объему, чем типовое, то возможно, стоит подумать о более элегантном решении.

2.5. Результат

Запуск корректно составленного решения должен привести к следующим определениям:

```
val isLeapYear = fn : int -> bool
val isCorrectDate = fn : int * int * int -> bool
val newStyleCorrection = fn : int * int * int -> int
val getNthInt = fn : int list * int -> int
val getNthStr = fn : string list * int -> string
val lastSmaller = fn : int * int list -> int
val firstNewMoonInt = fn : int * int * int -> int option
val firstNewMoon = fn : int * int * int -> (int * int * int) option
val dateToString = fn : int * int * int -> string
val isOlder = fn : (int * int * int) * (int * int * int) -> bool
val winterSolstice = fn : int -> int
val chineseNewYear = fn : int -> int * int * int
val chineseYear = fn : int -> string * string * string * string
val dateToChineseYear = fn : int * int * int -> string * string * string * string
val dateToAnimal = fn : int * int * int -> string
val animal = fn : string * (int * int * int) -> string
val extractAnimal = fn : (string * (int * int * int)) list * string -> (string * (int * int * int)) list
val extractAnimals = fn : (string * (int * int * int)) list * string list -> (string * (int * int * int)) list
val oldest = fn : (string * (int * int * int)) list -> string option
val oldestFromAnimals = fn : (string * (int * int * int)) list * string list -> string option
```