Лабораторная работа №1
«Люди — животные»

1. Предварительные сведения

Лабораторная работа посвящена отработке навыков

- работы с базовыми средствами функциональных языков программирования;
- написания программы в языке программирования со строгой статической явной типизацией;
- оперирования базовыми типами данных языка Standard ML;
- работы со составными типами данных: списки, кортежи, контейнеры Option;
- использования рекурсивных вычислений вместо конструкций цикла.

2. Общая схема разработки

Задания лабораторной работы посвящены вычислениям и преобразованиям календарных дат. Здесь рассмотрим связь календарей «Нового стиля» (Григорианского) и «Старого стиля» (Юлианского). Кроме того, мы обратимся к китайскому календарю для определения параметров года по китайскому летоисчислению (китайскому гороскопу).

Дату будем представлять как тройку (кортеж) целых чисел в которой на первом месте стоит номер дня в месяце, на втором — номер месяца, а на третьем — номер года.

Понятно, что не всякая тройка целых чисел может являться корректной датой. Мы будем считать корректной датой только тройки целых положительных чисел, в которых первое целое число может являться номером дня в месяце, второе — номером месяца, а третье — номером года в заданном типе календаря (юлианском и григорианском). Функции `isLongMonth`, `daysInMonth`, `isDayOK`, `isMonthOK`, `isCorrectDate` предназначены для проверки корректности даты.

Разница между юлианским и григорианским календарями в системе добавления лишнего дня февраля — в выборе високосного года. Функция `isLeapYear` как раз должна определять является ли год високосным, а функция `newStyleCorrection` должна вычислять накапливающуюся разницу в днях между календарями. Функции `toJulianDay`, `toGrigorianDay` должны осуществлять перевод от нового стиля к старому и от старого к новому.

Функции `incDateByNum` и `decDateByNum` — вспомогательные, они должны позволять увеличивать и уменьшать дату на определённое количество дней, а функция `younger` должна позволять сравнивать две даты.

Чтобы узнать значение знаков китайского гороскопа для заданного года, мы напишем функцию `chineseYear`. Наименования знаков этого гороскопа представлены в строковом виде в предопределённых списках. Для извлечения этих значений нам потребуется вспомогательная функция `getNthString`.

Но год по нашему календарю не совпадает с китайским и вычисленные знаки китайского гороскопа вступают в свои права не с 1 января нашего календаря, а с даты китайского Нового года. Поэтому, для правильного соответствия даты нашего календаря знакам китайского гороскопа, потребуется вычислить дату начала года по китайскому календарю. Эта дата зависит от дат новолуний и даты зимнего солнцестояния (дата китайского нового года — дата второго новолуния после даты зимнего солнцестояния в определённом регионе Китая). Поэтому нам потребуются функции для их вычисления.

Существует множество алгоритмов вычисления даты новолуния, отличающихся между собой различной степенью точности, из-за чего их применение может быть ограничено незначительным (в историческом масштабе) интервалом времени. Кроме того, стоит учитывать, что новолуние — момент времени, когда в разных часовых поясах разное время, а где-то и разные даты.

Мы будем вычислять дату новолуния (а, соответственно, и другие даты, привязанные к новолунию) по определённому упрощённому алгоритму для некоторого абстрактного часового пояса. Поэтому результат вычислений может отличаться от «справочных» (в редких случаях). Эти вычисления мы реализуем в функции `firstNewMoon`. Вычисления будут связаны с извлечением из списков предопределённых значений некоторых элементов, поэтому понадобятся вспомогательные функции для такого извлечения: `getNthFixed`, `numToDigits`, `listElements`, `listSum`, `maxSmaller`, `dateToCorrectionNums`.

Для вычисления даты зимнего солнцестояния напишем функцию `winterSoltice`.

Зная даты новолуний и зимнего солнцестояния мы сможем описать функцию `chineseNewYearDate` для вычисления даты китайского Нового года, а следовательно и функцию `dateToCineseYear` для установления соответствия между датой григорианского календаря и китайским гороскопом, функция `dateToAnimal` от знака китайского гороскопа оставляет только соответствующее животное.

Остальные реализуемые функции реализуют манипуляции с данными, содержащими даты. Обрабатываемые данные представляют собой «списки студентов»: списки кортежей, в которых первый элемент — строка, задающая имя студента, а второй — дата его рождения.

Функция `youngest` выбирает из списка студентов самого юного.

Функция `animal` определяет для одного студента (для одного кортежа) что он за «животное» по китайскому гороскопу. Функция `extractAnimal` извлекает из списка студентов всех, кто является заданным животным, а функция `extractAnimals` — всех, кто соответствует одному из животных в представленном списке.

Функция `oldStyleStudents` должна превращать список студентов с датами рождения по старому стилю в список студентов с датами рождения по новому стилю.

Функция `youngestFromAnimals` должна находить в списке студентов самое молодое «животное» из заданного списка животных, а функция `youngestFromOldStyleAnimals` должна делать то же самое, но для списка студентов с датами рождения, заданными по старому стилю.

Функция `listOfStringDates` должна превращать список студентов в список пар строк, в котором строковое представление даты заменяет тройку целых чисел. Для этого понадобится вспомогательная функция `dateToString` преобразования даты в строку.

Наконец функция `oldStyleStudentStringDates` должна переводить даты рождения в студентов, заданные по старому стилю, в строковые даты по новому стилю.

Таким образом, в ходе выполнения данной работы потребуется написать 35 функций на языке Standard ML.

3. Условности

В файле `lab-1-use.sml` приведены необходимые определения, для использования в лабораторной работе.

Во всех условиях задач лабораторной работы «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. Выражение «тройка чисел в формате даты» означает кортеж трёх целых чисел.

В файле `lab-1-use.sml` описан открытый модуль `MyDate` с минимальным функционалом для работы с такими датами. Во-первых, для простоты описания типа даты предварительно описан синоним типа `int * int * int`, который назван `date`. При типизации аргументов и результата функции следует использовать этот синоним вместо `int * int * int`. Кроме того, в модуле описаны функции `MyDate.anotherDay`, `MyDate.anotherMonth` и `MyDate.anotherYear`, принимающие на вход дату и целое число и выдающие новую дату, полученную из старой заменой соответственно номера дня, месяца или года на заданное число.

Так как тип данных `date` является синонимом кортежа трех целых чисел, то для элементов этого типа данных сохраняется весь арсенал функций и операций, применимых к таким кортежам.

В заданиях 20–22 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 такую подмену представлением в виде чисел «с фиксированной точкой».

Для того, чтобы визуально отделить использование в программе целых чисел от чисел с плавающей точкой в `lab-1-use.sml` с описан открытый модуль `Fixed` в котором введён синоним `fixed` для типа `int`. Здесь же определены функции для перевода целого числа в число с фиксированной точкой и обратно: `Fixed.fromInt` и `Fixed.toInt`.

Так как тип данных `fixed` является синонимом типа `int`, то для элементов этого типа данных сохраняются все средства для работы с целыми числами.

Помимо описаний модулей `MyDate` и `Fixed` в файле `lab-1-use.sml` заданы несколько списков для их использования в разрабатываемых функциях.

4. Замечания по выполнению заданий

4.1. 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

4.2. 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 4.1. Если вы считаете, что для выполнения какого-то из заданий необходима функция/конструкция, отсутствующая в разделе 4.1 — 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.

4.3. Предостережения насчёт 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, приведённой в задании.

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 насчёт 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.

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

5. Задания

Как говорилось выше, будем считать корректной датой тройки целых положительных чисел, в которых первое целое число может являться номером дня в месяце, второе — номером месяца, а третье — номером года в заданном типе календаря.

В задании 1 будем считать, что год по григорианскому и юлианскому календарю может принимать значения в диапазоне от 2 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. В юлианском календаре («по старому стилю») лишний день февраля добавляется если номер года делится на 4 без остатка. В григорианском календаре («по новому стилю») високосный год — год, номер которого делится на 400 без остатка или, если не кратен 100, то кратен четырём.

   Опишите функцию `isLeapYear`, первый аргумент которой — номер года, а второй — отметка о том, является ли календарь юлианским (`true`, если календарь юлианский и `false`, если григорианский). Функция должна возвращать `true`, если год с заданным номером високосный в указанном календаре, или `false` — в противном случае.

   Сигнатура итоговой функции: `int * bool -> bool`.

   Типовое решение — 3–4 строки кода.

2. Количество дней в отличных от февраля месяцах года нашего (григорианского) и юлианского календарей одинаково. Это количество — 30 или 31. Опишите функцию `isLongMonth` получающую номер месяца года (целое число от 1 до 12) и возвращающую `true`, если в соответствующем месяце 31 день.

   Сигнатура итоговой функции: `int -> bool`.

   Типовое решение — 2–4 строки кода.

3. Опишите функцию `daysInMonth`, получающую первым аргументом дату, а вторым — отметку о том, является ли календарь юлианским (`true`, если календарь юлианский и `false`, если григорианский). Функция должна возвращать количество дней в месяце заданной даты. Следует считать, что дата задана корректно.

   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`. Для проверки того, является ли месяц длинным, следует использовать функцию `isLongMonth`.

   Сигнатура итоговой функции: `date * bool -> int`.

   Типовое решение — 9 строк кода.

4. Опишите функцию `isDayOK`, получающую первым аргументом тройку чисел в формате даты, а вторым — отметку о том, является ли календарь юлианским. Функция должна возвращать `true`, если день месяца в указанной тройке чисел задан корректно. Следует считать, что месяц и год заданы корректно.

   Для вычисления количества дней заданного месяца следует использовать функцию `daysInMonth`.

   Сигнатура итоговой функции: `date * bool -> bool`.

   Типовое решение — 3–5 строк кода.

5. Опишите функцию `isMonthOK`, получающую тройку чисел в формате даты. Функция должна возвращать `true`, если номер месяца в указанной тройке чисел задан корректно. Функция не должна проверять корректность задания номера дня или года.

   Сигнатура итоговой функции: `date -> bool`.

   Типовое решение — 3–5 строк кода.

6. Опишите функцию `isCorrectDate`, получающую первым аргументом тройку чисел в формате даты, а вторым — отметку о том, является ли календарь юлианским. Функция должна возвращать `true`, если заданная тройка чисел представляет корректную дату, и `false` — в противном случае.

   Для проверки корректности номера дня и номера месяца следует использовать функции `isDayOK` и `isMonthOK` соответственно.

   Сигнатура итоговой функции: `date * bool -> bool`.

   Решение — 2–4 строки 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 9–11 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).

7. Опишите функцию `incDateByNum`, получающую три аргумента: дату, количество дней (неотрицательное целое число) и отметку о том, является ли календарь юлианским. Функция должна возвращать дату, наступающую после заданной через данное количество дней (то есть увеличить указанную дату на данное количество дней).

   Сигнатура итоговой функции: `date * int * bool -> date`.

   Типовое решение — 15 строк кода.

8. Опишите функцию `decDateByNum`, получающую три аргумента: дату, количество дней (неотрицательное целое число) и отметку о том, является ли календарь юлианским. Функция должна возвращать дату, наступившую до заданной за данное количество дней (то есть уменьшить указанную дату на данное количество дней).

   Сигнатура итоговой функции: `date * int * bool -> date`.

   Типовое решение — 17 строк кода.

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

   Опишите функцию `newStyleCorrection`, получающую григорианскую дату в качестве аргумента и возвращающую количество дней от неё до такой же даты 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 2021 года» функция должна возвращать `13`, а для «28 февраля 1700 года» — `10`.

   Сигнатура итоговой функции: `date -> int`.

   Объем решения — десяток строк.

10. Опишите функцию `toJulianDay` получающую григорианскую дату (дату «по новому стилю») и возвращающую соответствующую ей юлианскую дату (дату по «старому стилю»).

    Для высчитывания разницы в количестве дней между календарями на определённую дату следует использовать функцию `newStyleCorrection`. Если необходимо подсчитать увеличение/уменьшение даты на заданное количество дней, то для этого следует использовать функции `incDateByNum` / `decDateByNum` соответственно.

    Сигнатура итоговой функции: `date -> date`.

    Объем решения — 3–5 строк.

11. Опишите функцию `toGrigorianDay` получающую юлианскую дату (дату по «старому стилю») и возвращающую соответствующую ей григорианскую дату (дату «по новому стилю»).

    Для высчитывания разницы в количестве дней между календарями на определённую дату следует использовать функцию `newStyleCorrection`. Если необходимо подсчитать увеличение/уменьшение даты на заданное количество дней, то для этого следует использовать функции `incDateByNum` / `decDateByNum` соответственно.

    Сигнатура итоговой функции: `date -> date`.

    Объем решения — 3–5 строк.

12. Опишите функцию `younger`, получающую в качестве аргументов две даты и возвращающую `true`, если первая дата позже второй, или `false` — в противном случае.

    Сигнатура итоговой функции: `date * date -> bool`.

    Объем форматированного решения составит порядка 11–13 строк.

13. Опишите функцию `youngest`, получающую в качестве аргумента список имён студентов с датами рождения. Функция должна возвращать `NONE`, если заданный список пустой, или `SOME (name, d)`, если `name` — имя младшего из студентов в заданном списке, а `d` — его дата рождения. Если в заданном списке два или более

студентов имеют один и тот же возраст, то младшим из них будем считать студента, упоминающегося в списке последним.

Сигнатура итоговой функции: `(string * date) list -> string * date`.

Типовое решение — 11 строк.

14. Опишите функцию `getNthFixed`, получающую целое `n` и список чисел с фиксированной точкой. Функция должна вернуть `n`-й 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).

    Сигнатура итоговой функции: `int * fixed list -> fixed`.

    Форматированное решение занимает 3 строки.

15. Опишите функцию `numToDigits`, получающую целое число `num` и количество цифр `numDigits`, которое требуется получить из этого числа. Функция должна разбивать число `num` на цифры и возвращать в качестве результата список из `numDigits` последних цифр числа в обратном порядке. Например для `num` равного `25313` и `numDigits` равного `4` функция должна вернуть список `[3, 1, 3, 5]`.

    Если в числе количество цифр меньше чем `numDigits`, то необходимо добавлять к числу необходимое количество ведущих нулей.

    Сигнатура итоговой функции: `int * int -> int list`.

    Форматированное решение занимает 3 строки.

16. Опишите функцию `listElements` получающую в качестве аргументов список целых чисел и список списков чисел с фиксированной точкой. На каждой позиции i первого списка задан номер позиции j списка, расположенного на позиции i во втором списке. Функция должна извлекать из списков второго аргумента элементы из указанных в первом аргументе позиций, составлять из них список и возвращать его в качестве результата. Порядок элементов в итоговом списке должен сохраняться: элементы должны стоять на позициях списков, из которых они извлечены.

    Считаем, что все аргументы, включая номера позиций, заданы корректно.

    Сигнатура итоговой функции: `int list * fixed list list -> fixed list`.

    Форматированное решение занимает 3 строки.

17. Опишите функцию `listSum` получающую список чисел с фиксированной точкой и возвращающую сумму его элементов.

    Сигнатура итоговой функции: `fixed list -> fixed`.

    Типовое решение занимает 3 строки.

18. Опишите функцию `maxSmaller`, получающую список положительных чисел с фиксированной точкой и число с фиксированной точкой `amount`. 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 ноль.

    Сигнатура итоговой функции: `fixed list, fixed -> fixed`.

    В типовом решении 10 строк

В последующих заданиях, если не оговаривается иное, передаваемые даты должны являться датами по григорианскому календарю.

19. Для вычисления даты новолуния в заданном месяце нам понадобятся коэффициенты из списка `corrections` из файла `lab-1-use.sml`. Список `corrections` — список списков чисел с фиксированной точкой. В нем шесть списков. Для заданного месяца и года нужно извлечь по одному элементу из каждого списка в списке `corrections` для дальнейших вычислений.

    Опишите функцию `dateToCorrectionNums`, которая на основе заданной даты генерировала бы список из шести позиций соответствующих элементов списка `corrections`:

    - первый элемент результата — номер месяца в году, при условии, что нумерация ведётся от нуля (нулевой месяц — январь);
    - со второго по 5-й элементы — четыре цифры года, взятые в обратном порядке (сначала единицы, тысячи в конце);
    - последний элемент результата — остаток от деления номера года на 4.

Сигнатура итоговой функции: `date -> int list`.

В типовом решении 4–6 строк

20. Новолуния случаются примерно каждые 29.5 дней. День, в который случается новолуние, во многих календарях мира привязывается к дню начала месяца. В григорианском и юлианском календарях это не так, но даты новолуний всё равно остаются важными днями календаря, от которых ведётся отсчёт разных периодов, не связанных с обычным календарём напрямую (даты аграрного календаря, даты церковного календаря и пр.).

    Так как между двумя новолуниями примерно 29.5 дней, новолуния может не быть только в феврале. В любом другом месяце оно обязательно состоится, а иногда может случиться два новолуния в месяц.

    Опишите функцию `firstNewMoon`, получающую дату в качестве аргумента и возвращающую `NONE`, если в месяце заданной даты нет новолуния, и `SOME (fnum, d)`, если `fnum` — номер дня первого новолуния в месяце заданной даты в представлении с фиксированной точкой, а `d` — дата первого новолуния.

    Для получения результата сначала вычислим сумму значений с фиксированной точкой: сложим представленное в виде числа с фиксированной точкой количество дней разницы между юлианским и григорианским календарём на заданную дату с суммой извлечённых поправок из списка `corrections`, представленного в файле `lab-1-use.sml`.

    Позиции поправок в списке `corrections` определяются функцией `dateToCorrectionNums` от заданной даты. Но в передаваемой этой функции дате следует уменьшить номер года на единицу, если заданный месяц — январь или февраль.

    Из полученной суммы нужно вычесть максимально возможное число из списка `reductions`, значение которого меньше чем сумма уменьшенная на 1 (в формате с фиксированной точкой). Если такого числа в списке `reductions` не найдётся — оставить сумму без изменения.

    Полученное число `fnum` — номер дня заданного месяца, записанный в формате с фиксированной точкой. Если тройка `d`, составленная из этого числа, переведённого в целое, месяца и года, взятыми из исходной даты, будет являться корректной датой, то это и есть искомая дата новолуния. Тогда результатом функции будет контейнер `SOME`, содержащий кортеж из `fnum` и `d`.

    Если же корректной даты не получено, — результат функции `NONE`.

    Решение должно использовать описанные ранее функции `newStyleCorrection`, `listSum`, `listElements`, `dateToCorrectionNums`, `isCorrectDate`, `maxSmaller`, `Fixed.fromInt`, `Fixed.toInt`.

    Могут быть полезны функции `Fixed.anotherDay` `Fixed.anotherYear`.

    Сигнатура итоговой функции: `date -> (fixed * date) option`.

    В типовом решении 15 строк

В заданиях 21–35 будем считать, что год по григорианскому календарю может принимать значения в диапазоне от 201 до 2999 (проверять этот факт в решении не следует).

21. 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 отсчёта 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 дату дня зимнего солнцестояния.

    Все операции с дробными значениями следует проводить над числами с фиксированной точкой (т. е., например, сразу брать вместо 0.2422 значение 24220). Результат функции — целое число (следует превратить из числа с фиксированной точкой в целое число).

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

    Решение должно использовать функцию `Fixed.toInt`.

    Сигнатура итоговой функции: `int -> date`.

    Объем форматированного решения — пять строк.

22. 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 `chineseNewYearDate`, 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 `firstNewMoon` для декабря предыдущего года, и, на основании сравнения полученной даты с результатом функции `winterSolstice` для предыдущего года, отсчитывать нужную дату. В качестве точки отсчёта следует брать номер дня первого декабрьского новолуния в формате числа с фиксированной точкой (первый элемент результата `firstNewMoon`)

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

    Решение должно использовать описанные ранее функции `firstNewMoon`, `winterSolstice`, `younger`, `Fixed.toInt`.

    Сигнатура итоговой функции: `int -> date`.

    Типовое решение — 14 строк.

23. Опишите функцию `getNthString`, получающую целое `n` и список строк. Функция должна вернуть `n`-й 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).

    Сигнатура итоговой функции: `int * string list -> string`.

    Типовое решение — 3–4 строки.

24. Опишите функцию `dateToString`, выдающую для заданной даты строку в виде `"Month Day, Year"`, где `Month` английское наименование месяца, `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`. Наименования месяцев следует извлекать из представленного Вам списка `months` из файла `lab-1-use.sml`. 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.

    Сигнатура итоговой функции: `date -> string`.

    Решение — около 10 строк.

Для выполнения следующего задания потребуются списки `celestialChi`, `celestialEng`, `celestialColor`, `terrestrialChi`, `terrestrialEng`, представленные в файле `lab-1-use.sml`.

25. Летоисчисление в китайском календаре ведётся согласно 60-летнему циклу. Каждому году в цикле даётся название из двух частей.

    Первая часть — один из десяти представителей небесной ветви (небесные стихии): дерево как растение, дерево как материал, естественный огонь, высекаемый огонь, земля, керамика, руда, металл, текущая вода, стоячая вода. Китайские и английские наименования этих стихий представлены в списках `celestialChi` и `celestialEng` соответственно. Каждой паре представителей небесной ветви сопоставляется один из пяти цветов: зелёный, красный, коричневый, белый, чёрный. Наименования возможных цветов приведены в списке `celestialColor`.

    Вторая часть наименования года — один из двенадцати представителей земной ветви (земные стихии) представленные животными: крыса (мышь), корова (бык), тигр, кролик (заяц), дракон, змея, лошадь, овца, обезьяна, курица (петух), собака, свинья. 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 найдём остаток от деления полученного номера на 10, то получим порядковый номер небесной стихии. Если найдём целую часть от деления порядкового номера небесной стихии на 2, получим порядковый номер соответствующего цвета. Если найдём 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 четвёрку `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 четвёртая строка — английское наименование небесной стихии. Так, например, для года `1980` должен получиться результат `("Geng-Shen", "White", "Monkey", "Metal")`.

Сигнатура итоговой функции: `int -> string * string * string * string`.

Объем форматированного типового решения — 15 строк.

26. Опишите функцию `dateToChineseYear`, получающую дату в качестве аргумента и возвращающую, как и в предыдущим задании, четвёрку строк наименований для года китайского календаря, соответствующего заданной дате. Здесь следует обращать внимание на то, что дата китайского нового года не совпадает с датой нового года по григорианскому календарю.

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

    Сигнатура итоговой функции: `date -> string * string * string * string`.

    Решение составит порядка 8 строк.

27. 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.

    Сигнатура итоговой функции: `date -> string`.

    Решение — 2 строки.

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

    Сигнатура итоговой функции: `string * date -> string`.

    Форматированное типовое решение — 2 строки.

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

    Сигнатура итоговой функции: `(string * date) list * string -> (string * date) list`.

    Типовое решение — 5 строк.

30. 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 имён 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).

    Сигнатура итоговой функции: `(string * date) list * string list -> (string * date) list`.

    Решение — 4 строки.

31. Опишите функцию `youngestFromAnimals`, получающую два аргумента: первый — список имён студентов с датами рождения, второй — список строк — наименований животных из китайского календаря. Функция должна возвращать значение `string option: SOME (name, date)`, если `name` — имя младшего из студентов, которому по китайскому календарю соответствует какое-либо из животных заданного списка, а `date` — дата его рождения. Если в списке первого аргумента отсутствуют «животные» из второго аргумента, то результатом функции должно быть значение `NONE`.

    Сигнатура итоговой функции: `(string * date) list * string list -> (string * date) option`.

    Форматированное решение — 2–4 строки.

32. Опишите функцию `oldStyleStudents`, получающую список имен студентов с датами рождения по юлианскому календарю. Функция должна преобразовывать даты рождения к новому стилю. На выходе функции должен получаться список студентов, но с датами рождения по григорианскому календарю.

    Сигнатура итоговой функции: `(string * date) list -> (string * date) list`.

    Решение — 7–9 строк.

33. Опишите функцию `youngestFromOldStyleAnimals`, получающую два аргумента: первый — список имён студентов с датами рождения по юлианскому календарю, второй — список строк — наименований животных из китайского календаря. Функция должна возвращать значение `string option`: `SOME (name, date)`, если `name` — имя младшего из студентов, которому по китайскому календарю соответствует какое-либо из животных заданного списка, а `date` — дата его рождения по юлианскому календарю. Если в списке первого аргумента отсутствуют «животные» из второго аргумента, то результатом функции должно быть значение `NONE`.

    Сигнатура итоговой функции: `(string * date) list * string list -> (string * date) option`.

    Форматированное типовое решение занимает 14 строк.

34. Опишите функцию `listOfStringDates`, получающую список имён студентов с датами рождения. Функция должна преобразовывать даты рождения к строковому формату. На выходе функции должен получаться список студентов, но с датами рождения записанными на английском языке.

    Сигнатура итоговой функции: `(string * date) list -> (string * string) list`.

    Форматированное решение — 8 строк.

35. Опишите функцию `oldStyleStudentStringDates`, получающую список имён студентов с датами рождения по юлианскому календарю. Функция должна преобразовывать даты рождения студентов к григорианскому календарю и представить их в виде строки. На выходе функции должен получаться список студентов с датами рождения записанными на английском языке.

    Сигнатура итоговой функции: `(string * date) list -> (string * string) list`.

    Решение занимает 3 строки.

Общий объем решения, включая строки, предварительно заданные в файле, должен составить 440–460 строк.