электронный журнал
МОЛОДЕЖНЫЙ
НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО «Московский государственный технический университет им. Н.Э. Баумана»

УДК 004.021

Оптимизация алгоритмов исчисления дня года в среде LABVIEW

Студенты
кафедра «Информационные системы и телекоммуникации»: В. С. Попов, Е. Л. Мишина

Научный руководитель: О. Г. Петросян,
д. т. н., доцент кафедры «Информационные системы и телекоммуникации»

МГТУ им. Н.Э. Баумана
popov_vlad@mail.ru
gavric-cat@rambler.ru

Одна из самых популярных задач на всевозможных олимпиадах и соревнованиях по программированию – это перевод дня месяца и номера месяца в день с начала года. Например, 1 февраля – это 32-й день года. Очевидно, что у такой задачи должно быть три аргумента: день месяца, номер месяца и булева переменная, является ли год високосным или нет. Какие же способы решения существуют у этой задачи?

1. Решение с помощью условных операторов без вложенности

Самый простой и один из самых неэффективных алгоритмов для решения задачи перевода дня и номера месяца в день с начала года – алгоритм с применением условных операторов без вложенности. Такая программа действует следующим образом: если введённый месяц больше 1 (февраль и далее), то прибавить 31 (количество дней в январе) к результату, если больше 2 (март и далее), то прибавить 28 или 29 (в зависимости от того, високосный год или нет), если больше 3 (апрель и далее), то прибавить 31 (количество дней в марте), если больше 4 (май и далее), то прибавить 30 (количество дней в апреле) и т. д. В конце прибавить введённый день месяца.

Очевидные минусы такого подхода – лишние вычисления в условных операторах. Если пользователь в качестве месяца выбрал январь, то программа произведёт 11 лишних сравнений, а если был выбран декабрь, то лишних действий не окажется вовсе. В среднем такая программа делает 5 лишних сравнений.

2. Решение с помощью вложенных условных операторов

Проблему лишних проверок может решить вложенность условных операторов. Например, если номер месяца не больше 3, то осуществлять дальнейшие проверки не имеет смысла (алгоритм приведён на рис. 1).

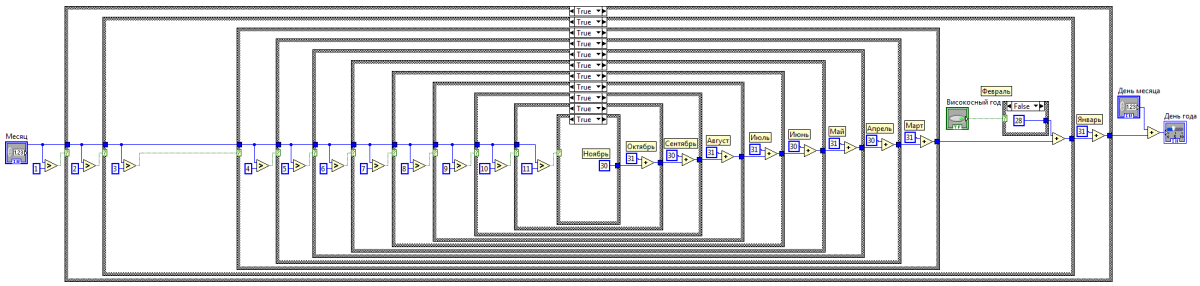

Рис. 1. Решение задачи в LabVIEW с использованием вложенных условных операторов. Все условные операторы, кроме первого, являются вложенными

Например, если номер месяца больше 10, то будет вызвана ещё одна проверка (рис. 2):

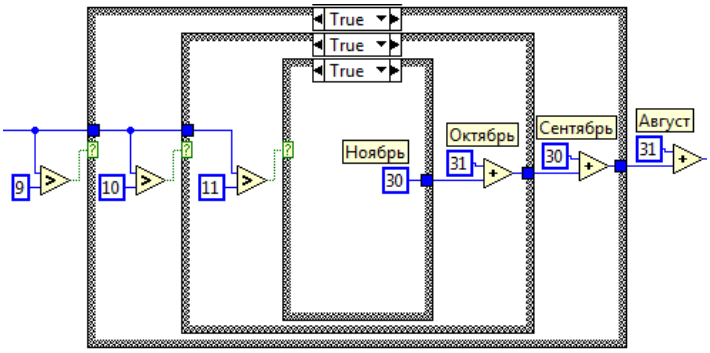

Рис. 2. Часть алгоритма наибольшей вложенности действует, если введён «большой» номер месяца

А если номер месяца меньше или равен 10, то этой проверки не произойдёт (рис. 3):

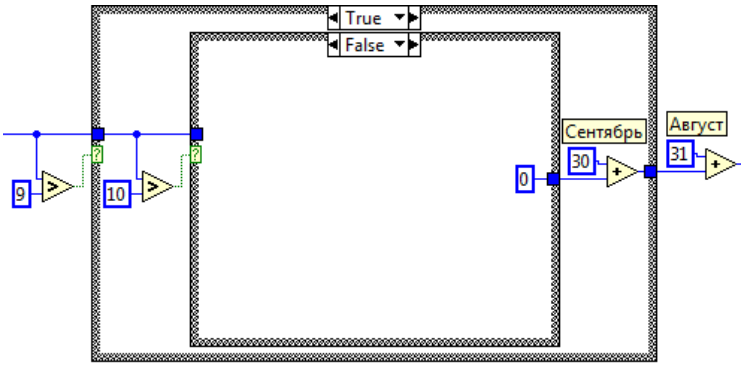

Рис. 3. Если номер месяца меньше либо равен 10, то количество дней в октябре и ноябре к ответу прибавлено не будет

Возможность управления потоком данных через большое количество вложенных условных операторов – далеко не лучшая сторона LabVIEW. Этот подход приводит к созданию громоздких, с трудом воспринимаемых инженером блок-диаграмм (впрочем, у традиционных языков программирования или у DSP-процессоров также существует эта проблема).

3. Решение с помощью цикла со счётчиком

Задачу нахождения дня года по дню и номеру месяца можно решить другим путём – через создание массива и использование цикла со счётчиком. В элементах массива будем хранить количество дней в месяце, номер которого соответствует номеру элемента массива. В зависимости от того, является ли год високосным или нет, необходимо

создавать разные массивы из-за того факта, что в феврале изменяется количество дней. Цикл проходит количество итераций, равное номеру месяца за вычетом единицы (так как количество дней в ведённом месяце суммировать не надо), на каждой итерации накапливается сумма дней в месяцах, предшествующих данному (к накопленной сумме прибавляется элемент массива, индекс которого равен номеру итерации цикла). В конце алгоритма к накопленной сумме прибавляется введённый день месяца (рис. 4).

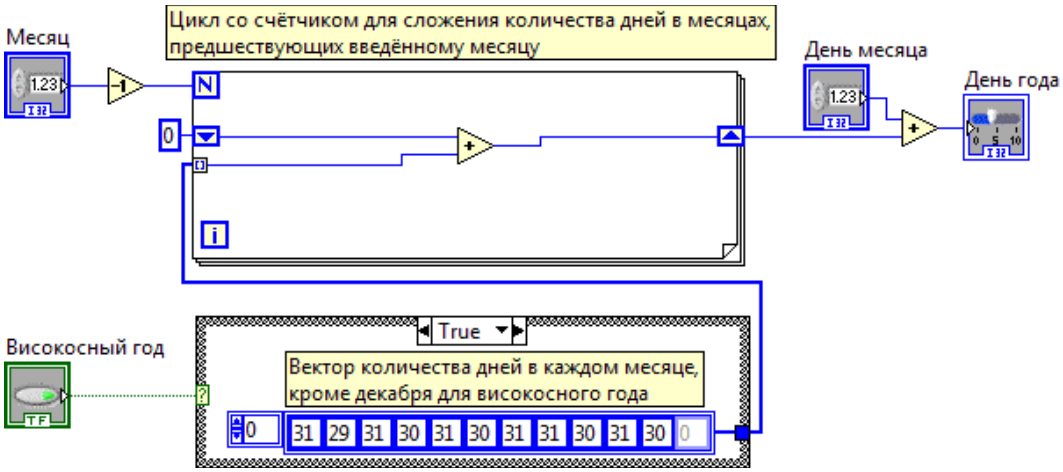

Рис. 4. Решение задачи в LabVIEW с помощью цикла со счётчиком и массива

Этот алгоритм получился гораздо лаконичнее, нежели алгоритм, базирующийся исключительно на операторах условия, на основе него удобно осуществлять проверку на корректность введённых данных, но он требует гораздо больше памяти из-за необходимости использовать массив и проигрывает по быстродействию.

4. Решение на основе чередования дней в месяцах

Каждый знает как подсчитать количество дней в месяце с помощью руки (рис. 5).

Рис. 5. Нахождение количества дней в месяце с помощью рук

Таким образом, дни в месяцах чередуются, кроме февраля и перехода между июлем и августом, чем можно воспользоваться. В алгоритм заложим, что количество дней в феврале равно 30, а перед тем, как выдать ответ, вычтем из ответа 1, если год високосный, или 2, если год невисокосный.

Тогда для подсчёта количества дней в году, если номер введённого месяца меньше либо равен 8 (8-й месяц – август), воспользуемся формулой 1:

$$31\cdot((m-1)\mathrm{div}2+(m-1)\bmod 2)+30\cdot((m-1)\mathrm{div}2)+d, \quad (1)$$

где m – введённый пользователем месяц,
d – введённый пользователем день,

div – операция целочисленного деления, mod – операция получения остатка от целочисленного деления.

Множитель $((m-1)\text{div}2+(m-1)\text{mod}2)$ формулы (1) равен количеству месяцев по 31 дню, идущих до введённого пользователем месяца (табл. 1):

Табл. 1

Номер месяца (m)	Количество месяцев по 31 дню до месяца m
1 (январь)	$(1-1)\text{div}2+(1-1)\text{mod}2=0$
2 (февраль)	$(2-1)\text{div}2+(2-1)\text{mod}2=1\text{div}2+1\text{mod}2=0+1=1$
3 (март)	$(3-1)\text{div}2+(3-1)\text{mod}2=2\text{div}2+2\text{mod}2=1+0=1$
4 (апрель)	$(4-1)\text{div}2+(4-1)\text{mod}2=3\text{div}2+3\text{mod}2=1+1=2$
5 (май)	$(5-1)\text{div}2+(5-1)\text{mod}2=4\text{div}2+4\text{mod}2=2+0=2$
6 (июнь)	$(6-1)\text{div}2+(6-1)\text{mod}2=5\text{div}2+5\text{mod}2=2+1=3$
7 (июль)	$(7-1)\text{div}2+(7-1)\text{mod}2=6\text{div}2+6\text{mod}2=3+0=3$
8 (август)	$(8-1)\text{div}2+(8-1)\text{mod}2=7\text{div}2+7\text{mod}2=3+1=4$

Множитель $((m-1)\text{div}2)$ формулы (1) равен количеству месяцев по 30 дней, идущих до введённого пользователем месяца, причём февраль принят за месяц с 30 днями (табл. 2):

Таблица 2

Номер месяца (m)	Количество месяцев по 30 дней до месяца m
1 (январь)	$(1-1)\text{div}2=0$
2 (февраль)	$(2-1)\text{div}2=1\text{div}2=0$
3 (март)	$(3-1)\text{div}2=2\text{div}2=1$
4 (апрель)	$(4-1)\text{div}2=3\text{div}2=1$
5 (май)	$(5-1)\text{div}2=4\text{div}2=2$
6 (июнь)	$(6-1)\text{div}2=5\text{div}2=2$
7 (июль)	$(7-1)\text{div}2=6\text{div}2=3$
8 (август)	$(8-1)\text{div}2=7\text{div}2=3$

Если номер введённого месяца больше 8 (сентябрь и далее), то из-за нарушения чередования между июлем и августом (7-й и 8-й месяц соответственно), воспользуемся формулой 2:

$$31\cdot((m-7-1)\text{div}2+(m-7-1)\text{mod}2+4)+30\cdot((m-7-1)\text{div}2+3)+d \quad (2)$$

Первые (левые) операнды операций целочисленного деления div и получения остатка от целочисленного деления mod изменили свой вид из-за необходимости «обойти» границу июля и августа, где чередования не происходит: теперь из номера месяца m необходимо вычесть не 1, а (7+1). Появление 7 в этой формуле (2) обусловлено тем, что все месяцы, идущие с января по июль, должны быть учтены иначе ввиду нарушения чередования. И именно для учёта этого изменения появилось третье слагаемое в множителе $((m-7-1)\text{div}2+(m-7-1)\text{mod}2+4)$, отражающем количество месяцев по 31 дню до введённого месяца, и второе слагаемое в множителе $((m-7-1)\text{div}2+3)$, отражающем количество месяцев по 30 дней до введённого месяца. В самом деле, с января по июль включительно есть 4 месяца по 31 дню и 3 месяца по 30 дней, если февраль принять как месяц с 30 днями.

Изменение $(m-1)$ на $(m-7-1)$ в формуле (2) можно изобразить иначе (рис. 6):

Рис. 6. Чередование дней в месяцах с августа по декабрь аналогично чередованию дней в месяцах с января по май

Множитель $((m-7-1)\text{div}2+(m-7-1)\bmod 2+4)$ формулы (2) равен количеству месяцев по 31 дню, идущих до введённого месяца (табл. 3):

Таблица 3

Номер месяца (m)	Количество месяцев по 31 дню до месяца m
9 (сентябрь)	$(9-7-1)\text{div}2+(9-7-1)\bmod 2+4=1\text{div}2+1\bmod 2+4=0+1+4=5$
10 (октябрь)	$(10-7-1)\text{div}2+(10-7-1)\bmod 2+4=2\text{div}2+2\bmod 2+4=1+0+4=5$
11 (ноябрь)	$(11-7-1)\text{div}2+(11-7-1)\bmod 2+4=3\text{div}2+3\bmod 2+4=1+1+4=6$
12 (декабрь)	$(12-7-1)\text{div}2+(12-7-1)\bmod 2+4=4\text{div}2+4\text{m}\text{od}2+4=2+0+4=6$

Множитель $((m-7-1)\text{div}2+3)$ формулы (2) равен количеству месяцев по 30 дней, идущих до введённого пользователем месяца (табл. 4):

Таблица 4

Номер месяца (m)	Количество месяцев по 30 дней до месяца m
9 (сентябрь)	$(9-7-1)\text{div}2+3=1\text{div}2+3=0+3=3$
10 (октябрь)	$(10-7-1)\text{div}2+3=2\text{div}2+3=1+3=4$
11 (ноябрь)	$(11-7-1)\text{div}2+3=3\text{div}2+3=1+3=4$
12 (декабрь)	$(12-7-1)\text{div}2+3=4\text{div}2+3=2+3=5$

К приведённым формулам осталось добавить пару условных операторов для составления программы: один условный оператор будет содержать приведённые формулы и условие, больше введённый месяц m восьми или же нет; второй условный оператор необходим для коррекции допущения, что в феврале 30 дней. Программа в среде LabVIEW выглядит следующим образом (рис. 7):

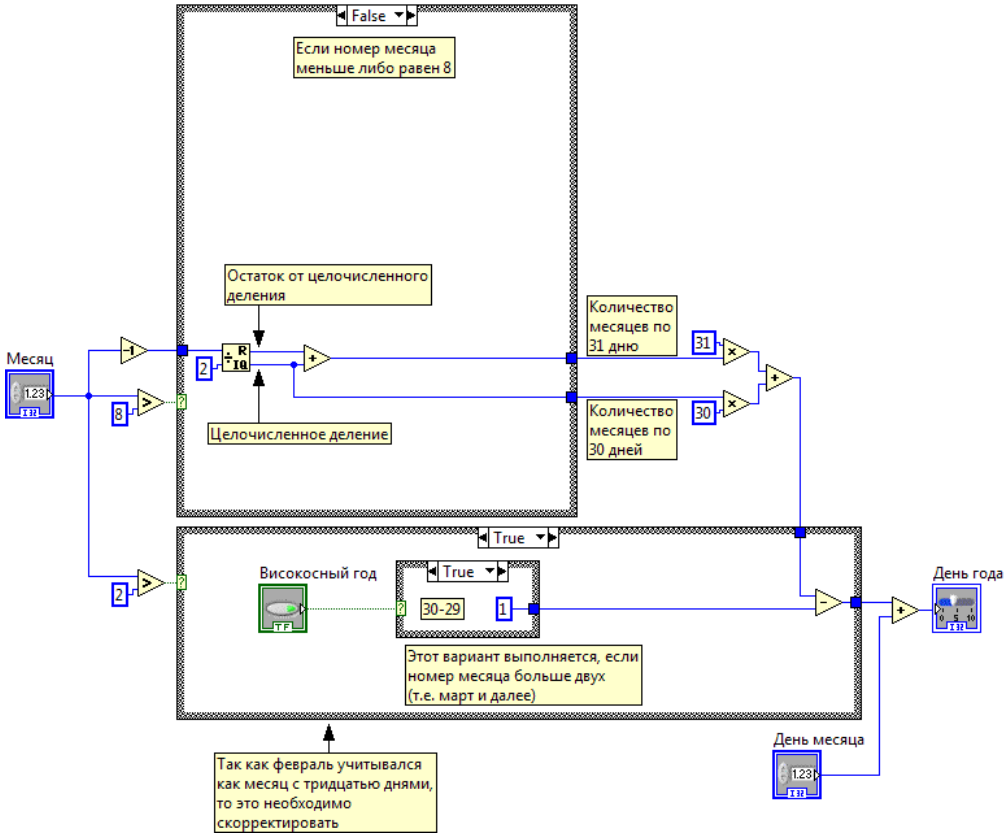

Рис. 7. Решение задачи в LabVIEW на основе чередования дней в месяцах

В случае високосного года (см. рис. 7) из результата будет вычитаться 1, в случае невисокосного года из результата нужно вычесть 2. Если введённый номер месяца меньше либо равен 8, то выполнится другая ветвь программы, так как необходимо изменить формулу для вычислений.

Испытания показали, что решение на основе чередования дней в месяцах является самым быстрым из всех приведённых. Решение с помощью цикла является самым медленным из-за необходимости поддержки функционирования этого цикла и необходимости выделения массива при каждом обращении к алгоритму.

В качестве приложений алгоритма можно привести определения срока годности продуктов, перевод дат между различными календарями, построение графиков по данным, содержащим даты и т.п.