

Разбор задачи “Дни рождения” Чемпионата
Республики Карелия по командному
программированию среди школьников.
Октябрь 2010.

Итак, напомним коротко условие задачи. Были заданы N Дней рождений. Для каждого непустого подмножества нужно было посчитать даты общих дней рождений. Напомним, что же такое общий День рождения из M человек. Для этого обозначим d_{ij} — сколько дней прожил i -ый человек в день с номером j (если дни нумеровать с момента рождения самого младшего человека из компании). Тогда день k считается общим днем рождения, если

$$\left\lfloor \frac{\sum_{i=1}^M d_{ik}}{365 \cdot M} \right\rfloor > \left\lfloor \frac{\sum_{i=1}^M d_{ik-1}}{365 \cdot M} \right\rfloor$$

От нас требовалось рассматривать только Дни рождения кратные 10 и не превышающие 100.

Во-первых, в данной задаче необходимо обработать входные данные. Дату в формате ДД.ММ.ГГГГ превратить, например, в два числа: дни с начала года и год. Удобней всего было завести структуру, у которой были бы записи о имени человека, годе и дне.

Во-вторых, алгоритмом решения этой задачи являлся перебор всех подмножеств из N элементов. Есть два подхода для реализации перебора всех подмножеств: рекурсивный и нерекурсивный.

Оба подхода хранят информацию о выбранных элементах в виде целого числа $mask$. Как же это, спросите Вы? А очень просто. Рассмотрим наше число в системе счисления с основанием два. Отсюда наше число запишется в таком виде: $mask = a_0 \cdot 2^0 + a_1 \cdot 2^1 + a_2 \cdot 2^2 \dots a_{N-1} \cdot 2^{N-1}$, где $a_i = 1$ или 0 — вошел или не вошел i -й элемент в наше подмножество. Очевидно, что максимальное значение $mask$, обозначающие все множество из N элементов, является числом $2^N - 1$.

Рассмотрим рекурсивный способ генерации всех подмножеств. Пусть есть функция rec от двух параметров $mask$ и num . Параметр num — номер текущего рассматриваемого элемента множества. Очевидно двойное неравенство $0 \leq num \leq N - 1$. Соответственно, как только num стал равен N , то мы уже сделали выбор, относительно каждого элемента множества (брать или не брать). Теперь, когда мы выбрали подмножество, нужно начать поиски общего Дня рождения. Об этом чуть позже.

Самая главная часть в рекурсивной реализации, это переходы внутри функции rec . Их два. Первый, это не включать текущий элемент в

рассматриваемое подмножество, второй — включать. Эти переходы эквивалентны следующим вызовам функции *rec*: $rec(mask + 2^{pos}, pos + 1)$ и $rec(mask, pos + 1)$.

Рассмотрим нерекурсивный способ генерации всех подмножеств. Он заключается в переборе всех значений числа *mask* от 1 до $2^N - 1$ (0 не рассматриваем, так как по условию задачи пустое подмножество не допустимо). Можно заметить, что каждое из значений *mask* будет каким-то подмножеством из *N* элементов, при том всегда новым.

Эти два метода различаются лишь порядком перечисления подмножеств. Ну и рекурсия, как известно, работает немного медленнее цикла, так как она требует дополнительных вызовов стека функций.

Хорошо, допустим мы получили некое число *mask*. Как теперь узнать, какие элементы в него включены, а какие нет? Необходимо всего лишь выполнить процедуру перевода числа *mask* в двоичную систему счисления (известный метод взятия остатка от 2 и деления на 2). Как только мы это выполнили, у нас есть список тех элементов, которые вошли в рассматриваемое сейчас подмножество. Напомним, что элемент подмножества — это переменная-структура, у которой есть поля год, дни, имя (*year*, *days*, *name*). Найдем в этом списке наибольший элемент (по значению $year \cdot 365 + days$), который соответствует самому младшему человеку. Теперь посчитаем разницу дней между Днем рождения самого младшего и остальных в подмножестве. Пусть *M* — количество элементов в нашем списке. И пусть они нумеруются с единицы. $\Delta_{sum} = \sum_{i=1}^M (days_{max} - days_i + (year_{max} - year_i) \cdot 365)$, где $days_{max}$ и $year_{max}$ — день и год рождения самого младшего человека. Теперь останется найти все решения уравнения $x \cdot M + \Delta_{sum} = age \cdot 365$, где *age* — количество лет, который каждый из подмножества прожил в среднем, *x* — количество дней, которое каждый из подмножества прожил с момента рождения самого младшего до текущего Дня рождения. Ответом будет дата рождения самого младшего + *x* дней. Уточним, что те решения, в которых $x \leq 0$ рассматривать не нужно, так как они не имеют смысла, с точки зрения задачи.

Сложность этого решения получается порядка $O(2^N \times N \times 10)$, т.е. вполне приемлемо для существующего ограничения по времени. Единственный нюанс для участников, использующих язык C++: чтение и вывод данных необходимо осуществлять с помощью функции `printf()` и `scanf()`.

Более подробно детали реализации можно посмотреть в решениях жюри.

*Разбор подготовил Басунков Владимир
Андреевич, преподаватель Клуба творче-
ства программистов ПетрГУ.*