Информатика

Гончаренко Валерий Евстафиевич
Доцент кафедры «Информационные технологии в экономике и организация производства» (ИТЭ и ОП) ГОУ ВПО «Ивановский государственный университет», кандидат технических наук. Ответственный организатор городской олимпиады школьников по информатике, член экспертной комиссии ЕГЭ по информатике и ИКТ.

Определение самых больших простых чисел

Повествование о простых числах, представленное в № 7 и 11 журнала «Потенциал» за 2010 г., можно считать незавершённым, пока не будут раскрыты вопросы определения самых больших простых чисел. На сегодняшний день самое большое простое число, известное человечеству, содержит в своей записи 12 978 189 десятичных цифр. В связи с такой огромной величиной числа возникает ряд вопросов:

- как формируют такие числа;
- как их хранят в памяти ПК;
- как над ними выполняются арифметические операции;
- как проверяется их принадлежность к простым числам.

Как формируют большие числа

Последовательность натуральных чисел бесконечна, и также бесконечна россыпь в них простых чисел. Неэффективно проверять каждое следующее число на простоту. Существуют различные способы формирования значения натуральных чисел, вероятность принадлежности которых к простым числам намного выше, чем у обычной последовательности чисел на числовой оси. Наибольшую известность и использование в определении самых больших простых чисел получили числа Мерсенна, которые вычисляются по формуле

$$M_p = 2^p - 1, \qquad (1)$$

где p – простое число.

У самого большого известного простого числа Мерсенна $p = 43\,112\,609$.

Названы эти числа в честь французского монаха, физика, математика Марена Мерсенна, родившегося в 1588 г. В «Физико-математических размышлениях» (1644) Мерсенн утверждал, что числа $2^p - 1$ являются простыми для $p = 2$, 3, 5, 7, 13, 17, 19, 31, 67, 127, 257. Этот факт произвёл сильное впечатление на его современников. Ведь оперировать такими огромными числами в те времена было делом немыслимым. Через сто лет Эйлер доказал простоту

числа M_{31}. И хотя, как выяснилось позже, в этом списке два числа – M_{67} и M_{257} – составные, можно лишь поражаться глубине проникновения Мерсенна в структуру таких чисел.

Келья монаха Мерсенна являлась центром общения и обмена информацией между выдающимися учёными всей Европы. По замечанию Блеза Паскаля (в свою очередь, язык программирования Pascal назван в его честь), монах ордена миноритов имел уникальный талант ставить новые научные проблемы,

но не разрешать их. Возможно, они так и не были бы решены, если бы он их не ставил. Именно этот талант и обусловил его миссию посредника в кругу самых знаменитых учёных того времени в Европе. Примечательно, что в одном из писем Р. Декарту Мерсенном были сформулированы более 30 вопросов, требующих разрешения. Основными результатами обширных научных исследований Мерсенна стали определение скорости распространения звука в атмосфере, результаты исследований по музыкальной акустике, была предложена схема зеркального телескопа. Умер Мерсенн 1 сентября 1648 г. после тяжёлой болезни и неудачной операции, но даже свою кончину он использовал во благо науки. По его распоряжению были проведены исследования, выявившие ошибку операции. Это были последние сведения, данные Мерсенном науке. На основе кружка Мерсенна (так называли регулярные встречи учёных в келье) была создана Парижская академия наук в 1666 г.

Как следует из формулы (1), усилия по формированию коллекции простых чисел, рассмотренные в №11 журнала «Потенциал», не являются напрасными и могут быть востребованы в формировании чисел Мерсенна при поиске очередного самого большого простого числа. Их в коллекции с лихвой хватит на многие годы вперёд, только в диапазоне натуральных чисел от 43 млн до 53 млн их около 560 тысяч. Число Мерсенна, которое по степени двойки отстоит от рекордсмена на 10 млн, а именно $M_p = 2^{53\,112\,617} - 1$, содержит в своей записи 15 988 491 десятичных цифр, для его распечатки на бумаге формата А4 потребуется 4750 страниц. В его записи первые и последние 13 цифр 7331835188054...9765671043071, возможно, это число тоже является простым числом, но это надо ещё доказать.

Как хранить большие числа в памяти ПК

Использование больших чисел в памяти ПК в информатике рассматривается в специальном направлении, получившем название «длинная арифметика». В настоящее время в науке и повседневной практике для отображения чисел используется позиционная десятичная система счисления, которая была разработана в Индии, но получила распространение и широкую известность благодаря арабским торговцам. Поэтому цифры и способ записи чисел стали называть арабскими. Это всем нам хорошо известные цифры: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Всего их десять,

поэтому система счисления и называется *десятичной*. Величина 10 называется основанием системы счисления, а самая большая цифра меньше основания на 1. Заметим, что «цифра» и «число» – это разные понятия, но в быту между ними чаще всего не видят разницы и отдают предпочтение слову «цифра». Даже из новостей мы можем слышать сообщение: «Окончательная цифра бюджета страны на следующий год ещё не установлена», хотя речь идёт явно о числе.

Любое число можно записать с помощью цифр, а для записи целого числа меньше десяти потребуется одна цифра. В дальнейшем будем рассматривать только целые числа. Одной и той же цифрой можно, например, записать число 555, в котором 5 сотен, 5 десятков и 5 единиц. От положения цифры в записи числа зависит величина, которую она отображает. Место в записи числа называется разрядом, и их счёт начинается с нуля, т. е. в нулевом разряде записываются единицы числа. Величины в смежных разрядах, записанные одинаковыми цифрами, отличаются в 10 раз.

Из рассмотренного порядка записи чисел в десятичной системе счисления вытекает простой и наглядный способ организации хранения больших чисел в памяти ПК. Достаточно запись числа представить в виде линейного массива с нумерацией его элементов от нуля и выше по необходимости. Нулевой элемент будет хранить количество единиц, первый – десятки, второй – сотни и так далее. Таким образом, от привычной записи числа его структура в линейном массиве будет «перевёртышем», т. е. цифры будут следовать в обратном порядке. Для современных ПК и систем программирования размещение в оперативной памяти нескольких линейных массивов длиной 100 млн элементов и более не составляет проблемы, что существенно больше для размещения известных самых больших простых чисел Мерсенна. То, что десятичные числа автоматически преобразуются в двоичный код, в рамках данной статьи можно опустить. В записи длинных чисел уже нет возможности проговаривать величины в различных разрядах, таких названий просто не существует. Неизменным остаётся соотношение в смежных разрядах, что будет использовано в арифметических операциях над ними.

Арифметические операции над длинными числами

Рассмотрим самый простой пример арифметической операции сложения двух чисел столбиком:

$$\begin{array}{r} 1234567890 \\ +\ 987654321 \\ \hline 2222222211 \end{array}$$

Они записываются друг под другом с условием выравнивания их с нулевых разрядов. Затем, начиная с разрядов единиц, определяется сумма цифр двух чисел и в итог записывается остаток от целочислен-

ного деления на 10, а целое количество десятков переносится в сумму следующего старшего разряда. Эта схема сложения без изменений повторяется многократно, пока не завершится записью цифры в самый старший разряд итога.

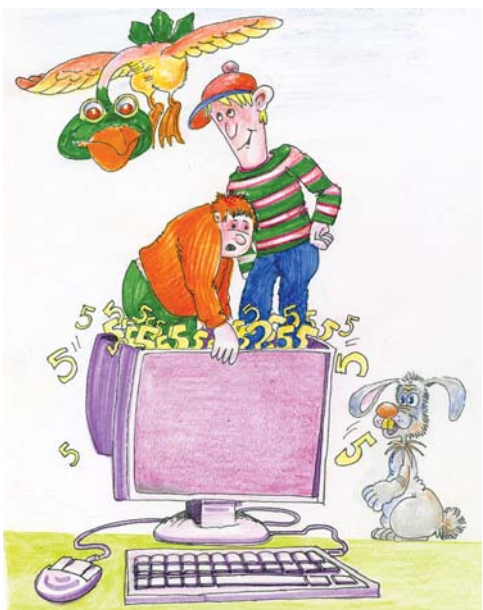
Ниже приводится демонстрационный пример программы на языке Pascal, реализующий арифметическую операцию сложения двух чисел, представленных в виде линейных массивов различной длины.

```
Program sum_a_b;
Uses Crt;
Const N=1002;
Type Tmas=array[0..N] of byte;
Var a,b,sum:Tmas;
    rega,regb,regsum:integer;
    s,p:byte;
    i:word;
begin
     clrscr;
     randomize;
     for i:=0 to N do
         begin
              a[i]:=0;
              b[i]:=0;
              sum[i]:=0;
         end;
     write('Введите разрядность первого слагаемого ');
     readln(rega);
     dec(rega);
     write('Введите разрядность второго слагаемого ');
     readln(regb);
     dec(regb);
     for i:=0 to rega do
         a[i]:=random(10);
     if (a[rega]=0) then a[rega]:=random(9)+1;
     for i:=0 to regb do
         b[i]:=random(10);
     if (b[regb]=0) then b[regb]:=random(9)+1;
     regsum:=-1;
     i:=0;
     p:=0;
     while ((i<=rega) or (i<=regb)) do
           begin
                s:=a[i]+b[i]+p;
                sum[i]:=s mod 10;
                p:=s div 10;
                inc(regsum);
                inc(i);
           end;
     if (p<>0) then
        begin
```

```
                inc(regsum);
                sum[regsum]:=p;
           end;
      writeln('a+b=sum');
      for i:=0 to rega do
          write(a[i]);
      writeln;
      for i:=0 to regb do
          write(b[i]);
      writeln;
      for i:=0 to regsum do
          write(sum[i]);
      writeln;
      writeln('Конец программы');
      readln;
end.
```

Данная программа приведена лишь в качестве демонстрации принципиальной возможности хранения чисел в памяти ПК в привычной последовательности записи десятичных цифр и выполнения арифметических операций над ними. Объём статьи не позволяет привести примеры всех арифметических операций, в том числе целочисленного деления (деления по модулю 10).

Будем считать, что проблемы программной реализации всех необходимых арифметических операций над длинными числами решены. Теперь необходимо получить запись числа Мерсенна, в котором показатель степени двойки может принимать значения порядка 40 млн. Самый простой алгоритм вычисления заданной степени двойки – это последовательное умножение на 2 начального значения, равного единице. Число умножений соответствует степени двойки. Эта идея реализована в приведённом ниже фрагменте программы на языке Pascal.



```
Program step2;
Uses Crt;
Const N=1000;
Type Tmas=array[0..N] of byte;
Var  m:Tmas;
     regm:word;
     st,i,k:word;
     p,s:byte;
```

```
begin
     clrscr;
     for i:=0 to N do
         m[i]:=0;
     m[0]:=1;
     write('Введите значение степени двойки st=');
     readln(st);
     regm:=0;
     for k:=1 to st do
         begin
                p:=0;
                for i:=0 to regm do
                    begin
                        s:=m[i]*2+p;
                        m[i]:=s mod 10;
                        p:=s div 10;
                    end;
                if (p<>0) then
                   begin
                       inc(regm);
                       m[regm]:=p;
                   end;
         end;
     write('2^',st,'=');
     for i:=regm downto 0 do
         write(m[i]);
     writeln;
     writeln('Конец программы');
     readln;
end.
```

В итоге, если ввести значение $st=10$, программа выдаст хорошо известное в информатике значение $2\text{^}10=1024$.

Для расширения диапазона вычисляемых значений необходимо в программах использовать не статические, а динамические линейные массивы, но при этом могут потребоваться большие затраты процессорного времени. Для ускорения вычислений можно воспользоваться свойством сложения степеней:

$$2^a \cdot 2^b = 2^{a+b}.$$

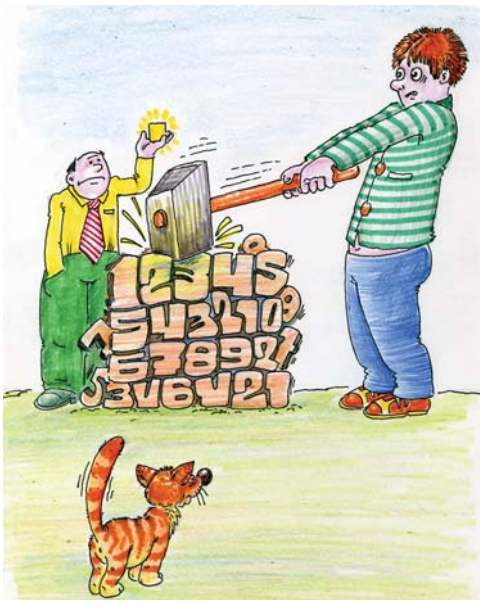
Например, если необходимо вычислить $2^{44\,000\,000}$, а ранее уже были вычислены значения $2^{43\,000\,000}$ и $2^{1\,000\,000}$, то достаточно перемножить эти два числа для получения искомого результата. Можно подвести итог, что нет принципиальных ограничений в вычислениях в понятиях длинной арифметики больших значений чисел Мерсенна. Проблема программной реализации заключается в разработке быстрых алгоритмов необходимых вычислений. Например, известен алгоритм быстрого умножения, предложенный в 1962 г. А.А. Карацубой. Понятно, что в конечном итоге имеет смысл использовать более современный и мощный язык программирования, например C++, в рамках которого автором были выполнены все рассмотренные вычисления.

Как проверить принадлежность числа Мерсенна к простым числам

Получение очередного значения числа Мерсенна – это всего лишь первый и незначительный по затратам процессорного времени результат в поиске очередного самого большого простого числа. Если читатель сформировал коллекцию простых чисел, то можно убедиться, какое большое количество их расположено хотя бы в ближайшем диапазоне исследований от 43 млн до 53 млн. И каждое значение M_p необходимо проверить на принадлежность к простым числам. Алгоритм решета Эратосфена здесь уже бессилен. Как отмечалось в одном из учебников по математике, «решето Эратосфена годится для поиска простых чисел Мерсенна не более чем консервная банка для плавания через Атлантический океан». Однако, отдавая должное заслугам великого грека, можно сказать, что решето Эратосфена стало парусом на фрегате французского математика Франсуа Люка, занимавшегося теорией чисел. Для проверки больших чисел существуют другие алгоритмы, которые подразделяются на вероятностные и детерминированные. В теории чисел простыми признаются лишь те числа, простота которых доказана детерминированными алгоритмами. Числам Мерсенна «повезло»: для доказательства их принадлежности к простым числам есть относительно нетрудоёмкий детерминированный алгоритм – тест Люка – Лемера.

Французский математик Франсуа Эдуард Анатоль Люка родился 4 апреля 1842 г. Важнейшие работы Люка относятся к теории чисел и неопределённому анализу. В 1878 г. Люка предложил критерий для определения того, простым или составным является число Мерсенна, и доказал простоту числа

$$M_{127} = 2^{127} - 1 = 170\ 141\ 183\ 460\ 469\ 231\ 731\ 687\ 303\ 715\ 884\ 105\ 727.$$



В течение 75 лет это число оставалось наибольшим простым числом Мерсенна, известным науке, но и по сей день оно остаётся самым большим, вычисленным «с карандашом в руке».

Не менее известен математик Люка своими изданиями занимательной математики. До сих пор задача, известная под названием «Ханойские башни», представляет интерес для математиков и программистов, проводятся даже международные семинары, посвящённые исключительно Ханойским башням.

В 1930 г. американский математик Д.Х. Лемер усовершенствовал критерий Люка, который впоследствии стал называться тест Люка – Лемера.

Расчёты по тесту Люка – Лемера заключаются в последовательных вы-

числениях критерия, начальное значение которого $S_0 = 4$. Каждое следующее значение критерия вычисляется по значению предыдущего значения и числа Мерсенна по рекуррентной формуле $S_m = (S_{m-1}^2 - 2) \bmod M_p$. Если значение последнего критерия равно нулю, т. е. $S_{p-2} = 0$, то число Мерсенна является простым числом. Например, для числа $M_7 = 2^7 - 1 = 127$:

$$S_0 = 4,$$
$$S_1 = (4*4 - 2) \bmod 127 = 14,$$
$$S_2 = (14*14 - 2) \bmod 127 = 67,$$
$$S_3 = (67*67 - 2) \bmod 127 = 42,$$
$$S_4 = (42*42 - 2) \bmod 127 = 111,$$
$$S_5 = (111*111 - 2) \bmod 127 = 0.$$

Легко проверить, что для следующего простого числа $p = 11$ тест Люка – Лемера не подтверждает простоту числа Мерсенна M_{11}.

Для больших значений степеней двойки p необходимо воспользоваться длинной арифметикой. Самым трудоёмким по затратам процессорного времени является деление по модулю M_p. Стандартными операциями div и mod уже не воспользуешься. Возникает проблема с большими затратами времени и ресурсов, но она всё-таки решается за счёт распределённых вычислений в сети. Существует широкомасштабный открытый интернет-проект по поиску простых чисел Мерсенна Great Internet Mersenne Prime Search (GIMPS), целью которого является поиск больших простых чисел Мерсенна путём распределённых вычислений на компьютерах любителей со всего земного шара. GIMPS выиграла денежный приз в 100 000 долл. США за нахождение простого числа из более чем 10 миллионов десятичных цифр $M_{43112609}$ и намеревается выиграть аналогичные призы в 150 000 и 250 000 долл. США, обещанные Electronic Frontier Foundation за нахождение простых чисел соответственно из более чем 100 и 1000 млн десятичных цифр.

Для проверки на простоту числа из более чем 100 млн десятичных цифр при текущем развитии вычислительной и алгоритмической техники потребуетя более трёх лет. Скорее всего, необходимо будет дождаться новых революционных прорывов в вычислительной технике или в математике до получения очередной премии.

Новости Новости Новости Новости Новости

Воздушный аккумулятор энергии

Необычный способ хранения энергии предполагается осуществить в США, где в штате Юта имеются соляные залежи. Вымыв соль, здесь планируют создать пещеры (на глубине 1,5 км) для хранения сжиженного воздуха. Это позволит гелиостанции, вырабатывающей в солнечные дни избыток электроэнергии, направлять его на производство и закачку в подземное хранилище жидкого воздуха. А при нехватке электроэнергии использовать воздух, переведя его в газообразное состояние, для вращения турбин электрогенератора. Таким образом, этот воздушный аккумулятор обеспечит создание полностью экологически чистой электростанции.