Материалы заданий заключительного этапа
Олимпиады школьников «Надежда энергетики»
по предмету «информатика» в 2013/2014 учебном году

Характер и уровень сложности олимпиадных задач направлены на достижение целей проведения олимпиады: выявить способных участников, твердо владеющих школьной программой и наиболее подготовленных к освоению образовательных программ технических ВУЗов, обладающих логикой и творческим характером мышления, умеющих алгоритмически описать реальные ситуации из различных предметных областей и применить к ним наиболее подходящие методы информатики. Необходимы знания способов описания алгоритмов (язык блок-схем, псевдокод) и умение работать с базовыми конструкциями.

Задания Олимпиады дифференцированы по сложности и требуют различных временных затрат на полное и безупречное решение. Они охватывают все разделы школьной программы, но носят, в большинстве, комплексный характер, позволяющий варьировать оценки в зависимости от проявленных в решении творческих подходов и продемонстрированных технических навыков. Участники должны самостоятельно определить разделы и теоретические факты программы, применимые в каждой задаче, разбить задачу на подзадачи, грамотно выполнить решение каждой подзадачи, синтезировать решение всей задачи из решений отдельных подзадач.

Успешное выполнение олимпиадной работы не требует знаний, выходящих за пределы школьной программы, но, как видно из результатов Олимпиады, доступно не каждому школьнику, поскольку требует творческого подхода, логического мышления, умения увидеть и составить правильный и оптимальный план решения, четкого и технически грамотного выполнения каждой части решения.

Умение справляться с заданиями Олимпиады по информатике приходит к участникам с опытом, который вырабатывается на тренировочном и отборочном этапах Олимпиады.

Олимпиада школьников «Надежда энергетики». Заключительный этап. Очная форма.
ЗАДАНИЕ ПО ИНФОРМАТИКЕ
РЕШЕНИЕ ВАРИАНТА 32111

Задание 1

На части поля треугольной формы с длинами сторон M, N, K метров посеяли укроп и сельдерей. Каждый побег с листьями занимает на поле фигуру, которая представлена в виде равностороннего треугольника с длиной стороны 10 см. К сожалению, летом информация о конкретном количестве посевов каждого вида была утеряна. Кроме того, часть посевов не взошла. Поэтому было принято решение о подсчёте «по факту», используя данные аэрофотосъёмки (по данным аэрофотосъёмки однозначно определяется, посев какого вида занимает конкретное место на поле). Вам необходимо разработать алгоритм и структуры хранения данных, чтобы посчитать количество посевов каждого вида.

Решение

При объявлении массива требуется указать количество элементов массива. Поскольку мы не знаем, как располагаются посевы относительно сторон треугольника, мы не можем подсчитать, сколько будет рядов и сколько посевов в каждом ряду, и поэтому не можем использовать массив для хранения информации. В таких случаях строят так называемые *списки*. **Список** – это динамически создаваемая структура из однотипных элементов, в которой каждый элемент хранит адрес следующего, а иногда также и предыдущего, элемента. Элементы списка, в отличие от элементов массива, не должны располагаться в памяти последовательно единым целым.

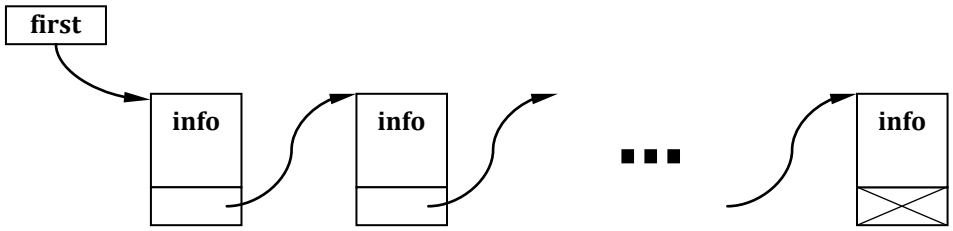


Рисунок 1. Однонаправленный список

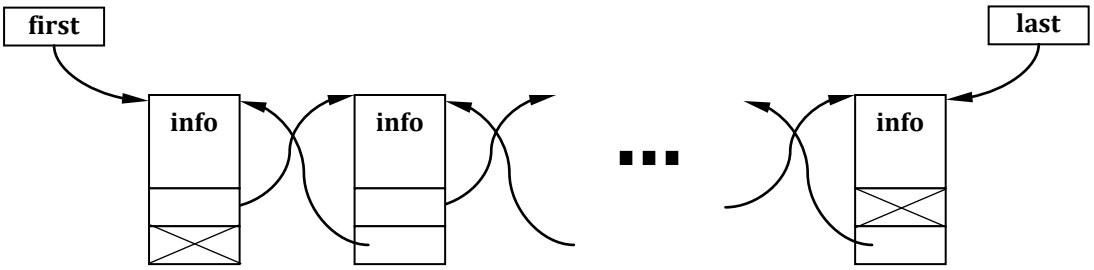


Рисунок 2. Двунаправленный список

Достоинства списков:

- размер списка ограничивается только доступным объёмом машинной памяти (массивы имеют ограничения на максимальный размер);
- при изменении последовательности элементов списка требуется не перемещение данных в памяти, а только коррекция адресов.

Недостатки списков:

- на адреса расходуется дополнительная память;
- доступ к элементам связной структуры может быть менее эффективным по времени, поскольку невозможно обратиться непосредственно к *i*-ому элементу списка – для того чтобы добраться до нужного элемента, надо пройти до него от начала списка.

Для решения данной задачи нам потребуется создать списки посевов, где каждый список будет соответствовать одному ряду, а элементом списка будет значение 0, если в данном месте ничего не выросло, 1 – если вырос укроп, и 2 – если вырос сельдерей. Все эти списки также объединим в список рядов. Будем считать, что для каждого списка определены следующие операции:

- создание пустого списка;
- добавление нового элемента в конец списка;
- установка текущего элемента на начало списка;
- переход к следующему элементу списка;
- удаление списка (вместе со всем содержимым).

```
алг Посевы (арг цел M, N, K)
нач
 цел dill, celery                        // Количество посевов укропа и сельдерея

 если M <= 0 или N <= 0 или K <= 0 то
  вывод 'Длины сторон треугольника должны быть положительными'
 иначе
  если M >= N + K или N >= M + K или K >= M + N то
   вывод 'Данные отрезки не могут составить треугольник'
  иначе

   создать список рядов
   для всех рядов
   нц
    добавить в список рядов новый элемент и сделать его текущим
    создать список посевов
    в текущий элемент списка рядов записать адрес только что созданного списка посевов
    для всех посевов в текущем ряду
    нц
     добавить в список посевов новый элемент и сделать его текущим
     в текущий элемент списка посевов добавить 0, 1 или 2 (по результатам аэрофотосъёмки)
    кц
   кц

   dill = 0
   celery = 0
   установить текущий элемент на начало списка рядов
   пока список рядов не закончился
   нц
    установить текущий элемент на начало списка посевов из текущего элемента списка рядов
    пока список посевов не закончился
    нц
     если текущее значение в списке посевов = 1 то
      dill = dill + 1
     иначе
      если текущее значение в списке посевов = 2 то
       celery = celery + 1
      всё
     всё
     перейти к следующему элементу списка посевов
    кц
    перейти к следующему элементу списка рядов
   кц

   вывод dill, celery
```

```
    пока список рядов не закончился
    нц
      удалить список посевов из текущего элемента списка рядов
    кц
    удалить список рядов
  всё
всё
кон
```

Задание 2

Серёжа интересуется планиметрией. Недавно он занялся вопросами построения правильных n-угольников. Прочитав дополнительную литературу, Серёжа узнал о теореме Гаусса-Ванцеля: правильный n-угольник можно построить с помощью циркуля и линейки тогда и только тогда, когда $n = 2^r \cdot p_1 \cdot p_2 \cdot \ldots \cdot p_k$, где r, k – натуральные числа, p_i – различные простые числа Ферма. Числа Ферма – числа вида $F_n = 2^{2^n} + 1$, где n – натуральное число или 0. Помогите юному геометру и разработайте алгоритм для определения того, можно ли построить правильный n-угольник. Учтите, что на 1 января 2014 г. найдено всего 5 простых чисел Ферма: 3, 5, 17, 257, 65537.

Решение

Поскольку числа Ферма – нечётные, то исходное значение n можно в первую очередь разделить на 2 несколько раз, пока результат будет оставаться целым, определив таким образом число r. Далее найдём числа Ферма, меньшие или равные $n / 2^r$, проверим, какие из них являются простыми, и постараемся выбрать из найденных чисел k таких, что их произведение будет равно $n / 2^r$.

Поскольку неизвестно, произведение скольких чисел Ферма даст нужный результат, необходимо организовать перебор всех возможных комбинаций. Т.е. если у нас есть m чисел Ферма, необходимо попробовать взять сначала каждое число, потом все возможные пары чисел, потом комбинации из 3, 4, …, $m - 1$, m чисел. Поскольку заранее неизвестно, чему равно число m, мы не можем использовать несколько вложенных циклов. Необходим другой способ перебора всех возможных комбинаций.

Как известно, числа в компьютере представляются в двоичной системе счисления. Эта система счисления использует всего две цифры – 0 и 1. Эти значения можно рассматривать как эквиваленты логических значений *ложь* и *истина*. Если нам надо перебрать все возможные комбинации из m чисел Ферма, то мы можем взять число, состоящее из m бит, и каждый бит этого числа поставить в соответствие одному из чисел Ферма. Если в числе на позиции с номером p стоит цифра 0, то соответствующее число Ферма не будет входить в комбинацию, а если на позиции с номером p стоит цифра 1, то соответствующее число Ферма будет входить в комбинацию. Число из m бит может представить 2^m значений. Значение 0 нам не нужно, т.к. комбинация, в которую ничего не входит, не представляет интереса. Остаётся $2^m - 1$ значений, что соответствует возможному числу комбинаций из 1, 2, …, $m - 1$, m чисел. Перебрать все возможные значения числа из m бит можно, начав со значения 1 и прибавляя единицу на каждом шаге.

Для примера рассмотрим перебор всех комбинаций из трёх чисел Ферма.

Число	**Двоичное представление**	**Используемые числа Ферма**
1	001	Первое
2	010	Второе
3	011	Первое и второе

4	100	Третье
5	101	Первое и третье
6	110	Второе и третье
7	111	Первое, второе и третье

Для проверки того, на каких позициях стоит цифра 1, используются так называемые *поразрядные операции*. Суть этих операций состоит в том, что в двух числах логические операции *И* и *ИЛИ* применяются к каждой паре бит, стоящих на одинаковых позициях. Например, 0011 and 0101 = 0001, 0011 or 0101 = 0111.

Когда у нас есть неизвестное число и надо определить, на каких позициях стоит цифра 1, берётся так называемая маска, равная 1, в которой есть только одна цифра 1. Операция *число* ***and*** *маска* даст значение, равное 0, если в *числе* на той позиции, на которой стоит цифра 1 в *маске*, стоит цифра 0, и значение, отличное от 0 в противном случае. После проверки одной цифры в *числе*, цифра 1 в *маске* смещается на следующую позицию. Для этого в компьютере реализована специальная операция сдвига, но можно использовать также операцию умножения на 2. Рассмотрим для примера проверку вхождения цифры 1 в число 5 (101 в двоичном представлении).

Маска	**Двоичное представление**	***Число* and *Маска***	**Интерпретация результата**
1	001	001	На первой позиции стоит цифра 1
2	010	000	На второй позиции стоит цифра 0
4	100	100	На третьей позиции стоит цифра 1

```
алг Многоугольник(арг цел N)
нач
  цел FermaNums[0..100]                    // Первое число получается возведением в степень 0
  цел PrimeFermaNums[0..100]               // Массив простых чисел Ферма
  цел r, cfn, cpfn, i, j mask, p
  лог possible

  ввод N

  r = 0
  пока N mod 2 = 0
  кц
    N = N div 2
    r = r + 1
  нц

  // Поиск чисел Ферма
  FermaNums[0] = 3
  cfn = 1
  пока FermaNums[cfn - 1] < N
  нц
    FermaNums[cfn] = (FermaNums[cfn - 1] - 1) * (FermaNums[cfn - 1] - 1) + 1
    cfn = cfn + 1
  кц

  // Выбор простых чисел Ферма
  cpfn = 0
  для i от 0 до cfn - 1
  нц
    если Тест_Пепина(FermaNums[i], i) то    // Тест на простоту для чисел Ферма
      PrimeFermaNums[cpfn] = FermaNums[i]
      cpfn = cpfn + 1
    всё
  кц
```

```
  //Поиск комбинации простых чисел Ферма, произведение которых даёт нужное значение
  possible = false
  i = 1
  пока i < 2 ^ cpfn – 1 and not result
  нц
    mask = 1
    p = 1
    для j от 0 до cpfn - 1
    нц
      если i and mask <> 0 то              // and – поразрядное И
        p = p * PrimeFermaNums[j]
      всё
     mask = mask << 1                      // << - операция сдвига,
    кц                                     // число 1 задаёт количество позиций для сдвига
    если p = N то
      possible = true
    всё
  i = i + 1
  кц

  если possible то
    вывод 'Данный многоугольник построить можно'
  иначе
    вывод 'Данный многоугольник построить нельзя'
  всё
кон

алг Тест_Пепина(арг цел F, N)                // F – число Ферма, N – его степень
нач
  цел b, i

  если i = 0 то
    вернуть истина
  всё

  b = 3
  для i от 1 до 2 ^ N – 1
  нц
    b = b * b mod F
  кц

  если b = F – 1 то
    вернуть истина
  иначе
    вернуть ложь
  всё
кон
```

Задание 3

На одной улице в ряд четыре дома, в которых живут Алексей, Егор, Виктор, Михаил. Известно, что **каждый из них владеет ровно одной профессией** Токарь, Столяр, Хирург, Окулист.

Известно, что

- Токарь живёт через дом от Столяра;
- Хирург живёт левее Токаря;
- Окулист живёт правее Токаря;
- Хирург живёт не рядом со Столяром;
- Михаил не Токарь;
- Алексей живёт рядом с Окулистом;
- Егор живёт справа от Токаря;
- Виктор рядом с Хирургом.

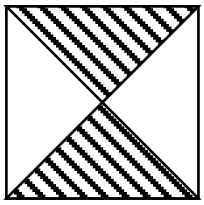
Кто какой профессией владеет?

Решение

Введем обозначения:
М – Михаил, В – Виктор, Е – Егор, А – Алексей;
Т – токарь, С – столяр, Х – хирург, О – Окулист.

Ответ: Х-М, Т-В, О-Е, С-А.

Задание 4



Дан двумерный массив целых чисел A[N, N]. Составить алгоритм, который за наименьшее число операций определяет наибольший элемент в заштрихованной области массива (см. рисунок).

Решение

Решение задачи может быть более оптимальным, если учесть различные способы хранения массивов в оперативной памяти. Существуют способы хранения по строкам и по столбцам. От выбранного способа зависит число операций при обращении и соответственно время доступа (выполнения).

```
алг Max(арг цел N, арг цел A[1..N, 1..N])
нач
  цел max, i, j

  max = A[1, 1]
  для i от 1 до (N + 1) div 2                  // При чётном N (N + 1) div 2 = N div 2
  нц                                           // При нечётном N берём центральный элемент массива
    для j от i до N – i + 1
    нц
      если A[i, j] > max то
        max = A[i, j]
      всё
      если A[N – i + 1, j] > max то
        max = A[i, j]
      всё
    кц
  кц

кон
```

Перебираются только нужные элементы массива в верхней половине, индексы элементов из нижней половины вычисляются из индексов верхней половины.

Задание 5

Школьники Пётр и Антон играли в бадминтон. Результаты игр записывали в таблицу (№ партии, результат Игрока 1, результат Игрока 2) на бумажном листе. Всего было проведено N игр ($1 \le N \le 50$). Кто чаще выигрывал? В качестве результата выведите имя игрока.

Решение

```
алг Бадминтон(агр цел N, арг цел table[1..50, 1..3])
нач
  цел i, petr, anton

  ввод N
```

```
  если N <= 0 то
    вывод 'Некорректное значение. N должно быть больше 0'
  иначе
    ввод table

    petr = 0
    anton = 0
    для i от 1 до N
    нц
      если table[i, 2] > table[i, 3] то
        petr = petr + 1
      иначе
        если table[i, 2] > table[i, 3] то
          anton = anton + 1
        всё
      всё
    кц

    если petr = anton то
      вывод 'Ничья'
    иначе
      если petr > anton то
        вывод 'Чаще выигрывал Пётр'
      иначе
        вывод 'Чаще выигрывал Антон'
      всё
    всё
  всё

кон
```

Задание 6

Задана логическая функция $F(A, B, C)$, значение которой равно 1 на наборах битов, определяемых двоичными представлениями чисел: $x_1 = 1_{10}$; $x_2 = 2_{10}$; $x_3 = 4_{10}$; $x_4 = 6_{10}$. Пожалуйста, запишите аналитически наиболее короткое представление данной функции в базисе (И, ИЛИ, НЕ).

Решение

Используя двоичную систему счисления, представим данные числа в двоичном коде:

$x_1 = 1_{10} = 001_2$;

$x_2 = 2_{10} = 010_2$;

$x_3 = 4_{10} = 100_2$;

$x_4 = 6_{10} = 110_2$.

Строим функцию *f*(*a*, *b*, *c*) в виде суммы трех произведений переменных *a*, *b*, *c* и их отрицаний. Каждое произведение имеет значение, равное 1 на наборе битов, определенных заданными числами, т.е.

	A	*B*	*C*	F	
$x_1 = 1_{10}$	0	0	1	1	$\overline{A} \cdot \overline{B} \cdot C$
$x_2 = 2_{10}$	0	1	0	1	$\overline{A} \cdot B \cdot \overline{C}$
$x_3 = 4_{10}$	1	0	0	1	$A \cdot \overline{B} \cdot \overline{C}$
$x_4 = 6_{10}$	1	1	0	1	$A \cdot B \cdot \overline{C}$

$$F(A,B,C)=\overline{A}\overline{B}C+\overline{A}B\overline{C}+A\overline{B}\overline{C}+AB\overline{C}=\overline{A}\overline{B}C+\overline{A}B\overline{C}+A\overline{C}=\overline{A}\overline{B}C+\overline{C}\left(\overline{A}B+A\right)=$$
$$=\overline{A}\overline{B}C+\overline{C}\left(A+B\right)=\overline{A}\overline{B}C+A\overline{C}+B\overline{C}$$

Ответ: $\overline{A}\overline{B}C+A\overline{C}+B\overline{C}$

Олимпиада школьников «Надежда энергетики». Заключительный этап. Очная форма.
ЗАДАНИЕ ПО ИНФОРМАТИКЕ
РЕШЕНИЕ ВАРИАНТА 32991

Задание 1

Дан пример на умножение

$$\begin{array}{r} 1\,2\,c_q \\ *\quad c\,a\,b_q \\ \hline 1\,1\,a\,2 \\ +\ b\,2\,b \\ \hline b\,b\,a\,a\,2_q \end{array}$$

Вам нужно определить значения q, a, b, c.

Решение

- отсутствует второе частичное произведение, о чем говорит сдвиг на два разряда третьей строки относительно второй, значит a = 0;

$$\begin{array}{r} 1\,2\,c_q \\ *\quad c\,0\,b_q \\ \hline 1\,1\,0\,2 \\ b\,2\,b \\ \hline b\,b\,0\,0\,2_q \end{array}$$

- второй разряд 1 + b = 0, значит был перенос в следующий разряд и перенос один, т.к. слагаемых два:

1 + b = q
1 + 1 + 2 = b b = 4, q = 5

$$\begin{array}{r} 1\,2\,c_5 \\ *\quad c\,0\,4_5 \\ \hline 1\,1\,0\,2 \\ 4\,2\,4 \\ \hline 4\,4\,0\,0\,2_q \end{array}$$

- 0, 1, 2, 4 уже присутствуют в примере, значит c = 3, т.к. весь алфавит не более 4 при q = 5. Проверка: c*c = 4 + q (перенос), $c^2 = 9$, c = 3.

Ответ: q = 5, a = 0, b = 4, c = 3

Задание 2

До принятия международной системы единиц СИ в физике применялась система СГС (сантиметр-грамм-секунда). Таким образом, например, сила измерялась в динах (Одна дина численно равна силе, которая сообщает телу массой в 1 грамм ускорение в один сантиметр в секунду за секунду). Мощность измерялась в эргах/с (1 эрг равен работе силы в 1 дин при перемещении точки приложения силы на расстояние 1 см в направлении действия силы). Вам попались записи в системе СГС. Пожалуйста, разработайте алгоритм, позволяющий перевести заданную величину в современную систему СИ. Переводу подлежат: единица длины, единица массы, единица времени, единица силы, единица работы, единица мощности.

Решение

Для единицы длины пользователю необходимо использовать обозначение «см», единицы массы – «г», единицы времени – «с», единицы силы – «дин», единицы работы – «эрг», единицы мощности – «эрг/с». При вводе каких-либо других сочетаний символов будет выводиться сообщение «Неизвестное обозначение».

```
алг Единицы_измерения (арг вещ value, строка sign)
нач
  вещ result

  ввод value, sign

  если sign = 'см' то
    result = value * 1e-2
    вывод result, ' м'
  иначе
    если sign = 'г' то
      result = value * 1e-3
      вывод result, ' кг'
    иначе
      если sign = 'с' то
        result = value
        вывод result, ' с'
      иначе
        если sign = 'дин' то
          result = value * 1e-5
          вывод result, ' Н'
        иначе
          если sign = 'эрг' то
            result = value * 1e-7
            вывод result, ' Дж'
          иначе
            если sign = 'эрг/с' то
              result = value * 1e-7
              вывод result, ' Вт'
            иначе
              вывод 'Неизвестное обозначение'
            всё
          всё
        всё
      всё
    всё
  всё
кон
```

Задание 3

Юный математик Пётр увлекается поиском различных чисел, названных в честь выдающихся математиков. Недавно он узнал, что числа Мерсе́нна – числа вида $M_n = 2^n - 1$, где n – натуральное число. Пётр захотел проверить числа M_n на простоту. Помогите ему и разработайте соответствующий алгоритм.

Решение

```
алг Числа_Мерсенна (арг цел N)
нач
  цел M, i

  ввод N

  для i от 1 до N
  нц
```

```
    M = 2 ^ i – 1
    если Простое(M) то
      вывод 'Число Мерсенна ', M, ' является простым'
    иначе
      вывод 'Число Мерсенна ', M, ' не является простым'
    всё
  кц

кон
```

Можно вычислять n-ое число Мерсенна через $(n-1)$-ое. Тогда основной алгоритм примет следующий вид.

```
алг Числа_Мерсенна (арг цел N)
нач
  цел M, i

  ввод N

  M = 1
  для i от 1 до N
  нц
    если Простое(M) то
      вывод 'Число Мерсенна ', M, ' является простым'
    иначе
      вывод 'Число Мерсенна ', M, ' не является простым'
    всё
    M = (M + 1) * 2 – 1
  кц

кон
```

Что касается вспомогательного алгоритма проверки чисел на простоту, может быть несколько вариантов.

Первый вариант – примитивный перебор всех чисел.

Второй вариант – отбрасываем чётные числа.

```
алг Простое (арг цел N)
нач
  цел i

  если N = 1 то
    вернуть ложь                                  // Число 1 не считается простым
  всё
  если N = 2 то
    вернуть истина                                // Число 2 является простым
  всё
  если N mod 2 = 0 то
    вернуть ложь                                  // Число, которое делится на 2, – не простое
  всё

  для i от 3 до целая_часть(sqrt(N)) шаг 2
  нц
    если N mod i = 0 то
      вернуть ложь
    всё
  кц
  вернуть истина

кон
```

Третий вариант – отбрасываем чётные числа и числа, кратные 3.

```
алг Простое (арг цел N)
нач
  цел i

  если N = 1 то
    вернуть ложь
  всё
  если N = 2 или  N = 3 или  N = 5 или  N = 7 то
    вернуть истина
  всё
  если N mod 2 = 0 то
    вернуть ложь
  всё
  если N mod 3 = 0 то
    вернуть ложь
  всё

  для i от 1 до целая_часть(sqrt(N)) div 6 + 1
  нц
    если N mod (6 * i – 1) = 0 то
      вернуть ложь
    всё
    если N mod (6 * i + 1) = 0 то
      вернуть ложь
    всё
  кц
  вернуть истина

кон
```

В приведенных выше вариантах число выполняющихся операций уменьшается от первого к третьему. Для участников, заинтересовавшихся дальнейшей оптимизацией решения, мы рекомендуем ознакомиться в литературе со специальным тестом Люка-Лемера, ориентированным на выявление простоты чисел Мерсенна.

Задание 4

При обработке строки текста АБВЕГД был применен следующий алгоритм: если четвертая буква строки согласная, то меняем местами первую и четвертую буквы, а если гласная, то переносим ее на вторую позицию в строке. Запишите последовательность, получившуюся после пятикратного применения этого алгоритма.

Решение

АБВЕГД – исходное состояние
АЕБВГД – шаг 1
ВЕБАГД – шаг 2
ВАЕБГД – шаг 3
БАЕВГД – шаг 4
ВАЕБГД – шаг 5

Ответ: ВАЕБГД

Задание 5

Для ведения наблюдений за погодой в марте 2013 г. в г. Сочи метеорологами была составлена на листе бумаги таблица. Для каждого дня в марте было записано значение температуры (диапазон [–20; +20]). Разработать алгоритм определения среднего значения температуры в дни, когда температура была положительна?

Решение

```
алг Средняя_температура(арг вещ temperature[1..31])      // В марте 31 день
нач
  вещ s
  цел i, k

  ввод temperature

  s = 0
  k = 0
  для i от 1 до 31
  нц
    если temperature[i] > 0 то
      s = s + temperature[i]
      k = k + 1
    всё
  кц

  если k = 0 то
    вывод 'Не было дней с положительной температурой'
  иначе
    s = s / k
    вывод 'Средняя температура в дни, когда температура была положительна, равна ', s
кон
```

Задание 6

Двузначное десятичное число в сумме с числом, записанном теми же цифрами, но в обратном порядке, даёт квадрат некоторого двузначного числа. Составить алгоритм, определяющий эти числа.

Решение

Первый вариант – примитивный. Перебираем все двузначные числа, кроме тех, которые делятся на 10, и проверяем, что получится.

```
алг Двузначные_числа()
нач
  цел num, d1, d2, num2

  для num от 11 до 99
  нц
    если num mod 10 <> 0 то
      d1 = num div 10
      d2 = num mod 10
      num2 = d2 * 10 + d1
      если целая_часть(sqrt(num + num2)) * целая_часть(sqrt(num + num2)) = num + num2 то
        вывод num, num2
      всё
    всё
  кц

кон
```

Второй вариант – рассудительный. Пусть a и b – цифры числа. Тогда первое число равно $10a + b$, второе – $10b + a$, а их сумма $(10a + b) + (10b + a) = 11(a + b) = x^2$. Из этого следует, что квадрат третьего числа x должен делиться на 11. Поскольку 11 – простое число, мы не можем разложить его на сомножители, тем более на одинаковые сомножители, которые могли бы входить в само число x (например, если мы знаем, что квадрат некоторого числа делится на 25, то можно сделать вывод, что само число, как минимум, делится на 5). Следовательно, число x – это 11, или 22, или 33 и т.д. Но 22 * 22 = 484, что гораздо больше, чем возможная сумма двух двузначных чисел. Значит, $x = 11$, а $x^2 = 121 = 11(a + b)$. Следовательно, сумма цифр исходного числа равна 11. Цифры 0 и 1 не подходят, т.к. нет цифр, которые в сумме с ними давали бы 11. Таким образом, одна из цифр берётся из диапазона от 2 до 9, а вторую надо подобрать так, чтобы сумма равнялась 11.

```
алг Двузначные_числа()
нач
  цел d1, d2

  для d1 от 2 до 9
  нц
    d2 = 11 – d1
    вывод d1 * 10 + d2, d2 * 10 + d1
  кц

кон
```

Мы берём только одно число и на его основе получаем второе, поэтому пары (29, 92) и (92, 29) можно рассматривать как разные.

Задание 7

Найдите позиционную систему счисления, в которой правильны следующие равенства
а) 12 + 22 = 100
б) 11 / 110 = 0.1
в) 70 – 1 = 68

Решение

Задания а) и в) могут решаться методом направленного перебора. Для случая а) перебор начинается с 3, поскольку в условии присутствует цифра 2. Для случая в) перебор ограничен: сверху значением 9, снизу – значением 8.

Однако такой способ решения неэффективен.

Вместо этого следует решать задания так:

а) $12_q + 22_q = 100_q$. Раскладывая в ряд по полиномам, получим $1 \cdot q^1 + 2 \cdot q^0 + 1 \cdot q^1 + 2 \cdot q^0 = 1 \cdot q^2 + 0 \cdot q^1 + 0 \cdot q^0$

Отсюда получаем следующее уравнение для десятичной системы счисления: $q^2 = q + 2 + 2 \cdot q + 2$. Решая уравнение, получаем q=4.

в)$70_q – 1 = 68_q$

7q-1 = 6q+8

q=9

Для случая б) подходит любая позиционная система счисления.

Ответ:
а) 4
б) любая позиционная система счисления
в) 9