ЗАДАНИЕ ПО ИНФОРМАТИКЕ
ВАРИАНТ 73991 для 9 класса

Для заданий 1, 2, 4, 5 требуется разработать алгоритмы на языке блок-схем, псевдокоде или естественном языке

1. Для проверки, является ли большое целое простым, может использоваться вероятностный тест Ферма. Пусть $p > 2$ – проверяемое число. Тогда:

   - случайно выбираем a: $2 \le a \le p - 2$;
   - если $a^{p-1} \neq 1 \ \{\mathrm{mod}\ p\}$, то p – составное.

   В тесте Ферма эти проверки выполняются для t случайно выбираемых a.

   Написать алгоритм проверки вводимого числа на простоту по тесту Ферма.

   Примечание: $x = y \ \{\mathrm{mod}\ n\}$, если существует целое k, для которого $x = y + k \bullet n$.

Решение. В цикле t раз выбираем число a в заданном диапазоне. Для каждого a проверяем условие $a^{p-1} \neq 1 \ \{\mathrm{mod}\ p\}$. Для того чтобы при возведении в степень не получилось слишком большого числа (выходящего за пределы чисел, представимых в компьютере), можно применять операцию вычисления остатка от деления после каждого умножения на a. Если для какого-то значения a результат не равен 1, то число – составное, и проверку можно прекращать.

Выполняется следующее равенство: $(x \cdot y) \bmod p = ((x \bmod p) \cdot (y \bmod p)) \bmod p$

Поскольку в нашем случае $a < p$, то $a \bmod p = a$. Значит, на первом шаге цикла мы имеем результат операции $a^n \bmod p$ (для $n = 1$). Тогда $a^{n+1} \bmod p = ((a^n \bmod p) \cdot (a \bmod p)) \bmod p$. $a^n \bmod p$ мы уже имеем, и $a \bmod p = a$. Остаётся только выполнить умножение и взятие остатка от деления.

```
алг ТестФерма()
нач
  цел p, a, t, i, r
  лог prime

  ввод p, t
  если p <= 2 или t <= 0 то
    вывод "Некорректные исходные данные"
  иначе

    prime = истина
    i = 1
    пока i <= t и prime
    нц
      генерация a в диапазоне от 2 до p - 2
      r = a
      для j от 2 до p – 1
      нц
        r = (r * a) mod p
      кц
      если r <> 1 то
        prime = ложь
      всё
    кц

    если prime то
      вывод "Число простое"
    иначе
      вывод "Число составное"
    всё
```

```
  всё
кон
```

2. В доме у Николая есть длинная наклонная лестница с большим числом крупных ступеней L. На ступенях сверху-вниз любят прыгать дети со двора. На каждой ступеньке нарисован вес – натуральное число. Николай спустился по лестнице прыжками. Прыгать можно только на 1, 3 или 4 ступеньки. Каков суммарный вес ступенек, по которым спустился Николай?

Решение. Пусть у нас есть массив из L элементов, в которых записан вес каждой ступени. Запишем результат спуска Николая в виде массива из L элементов, причём элемент равен 1, если Николай наступал на данную ступеньку, и 0 – в противном случае. Для проверки соблюдения правил будем вычислять разницу между индексами элементов, содержащих значение 1. Для этого положим сначала номер предыдущей использованной ступени 0, затем при нахождении в массиве значения 1 вычисляем разницу. Если она не равна 1, 3 или 4, значит, правила была нарушены. Иначе прибавляем к общему результату вес текущей ступени и запоминаем номер текущей ступени как номер предыдущей.

```
алг Спуск()
нач
  цел l, weights[l], steps[l], i, prev, res
  лог right

  ввод l
  если l <= 0 то
    вывод "Некорректные исходные данные"
  иначе

    для i от 1 до l
    нц
      ввод weights[i]
    кц
    для i от 1 до l
    нц
      ввод steps[i]
    кц

    res = 0
    right = истина
    prev = 0
    i = 1
    пока i <= l и right
    нц
      если steps[i] = 1 то
        если i – prev = 1 или i – prev = 3 или i – prev = 4 то
          res = res + weights[i]
          prev = i
        иначе
          right = ложь
        всё
      всё
    кц

    если right то
      вывод "Итоговый вес равен ", res
    иначе
      вывод "Николай нарушил правила"
    всё

  всё
кон
```

3. Школьник Сережа любит играть с калькулятором. Он часто сначала делит вещественные числа a и b друг на друга, а затем результат умножает на b. Выполнив эти действия много раз (сначала много делений, а затем столько же умножений), Сережа получил в результате не исходное число. Объясните, почему?

Решение. Возникающая в процессе вычислений погрешность, обусловленная округлением вещественных чисел из-за ограниченного количества знаков в числе при представлении в ЭВМ, изменяет результат.

4. Не используя дополнительный массив или простые методы сортировок, найти в одномерном массиве номера трех первых минимальных элементов.

Решение. Найдём минимальный элемент массива. Далее проходим по массиву, ищем элементы, равные минимальному, и переставляем их в начало массива. Если в массиве есть три равных минимальных значения, значит, поиск можно закончить. Если же таких значений меньше трёх, надо снова найти минимальное значение, не рассматривая первые один или два элемента массива.

Приведённый алгоритм может быть использован для поиска любого количества минимумов.

```
алг ТриМинимума()
нач
  цел n, x[n], min, k, im, i

  ввод n
  если n <= 0 то
    вывод "Некорректные исходные данные"
  иначе

    для i от 1 до n
    нц
      ввод x[i]
    кц

    k = 1
    пока k <= 3
    нц
      // Находим минимальный элемент и его номер
      min = x[k]
      im = k
      для i от k + 1 до n
      нц
        если x[i] < min то
          min = x[i]
          im = i
        всё
      кц

      // Переставляем найденный минимальный элемент на k-ое место
      x[im] = x[k]
      x[k] = min

      // Ищем элементы, равные минимальному, после минимального, считаем их и переставляем
      для i от im + 1 до n
      нц
        если x[i] = min то
          k = k + 1
          x[i] = x[k]
          x[k] = min
        всё
      кц
      k = k + 1
    кц

    для i от 1 до 3
    нц
      вывод x[i]
    кц

  всё
кон
```

5. Числа Фибоначчи – натуральные числа, удовлетворяющие следующим соотношениям: $F_0 = 1,\ F_1 = 1,\ F_n = F_{n\text{-}1} + F_{n\text{-}2},\ n \geq 2.$ Период Пизано $\pi(m)$ – это длина периода последовательности Фибоначчи по модулю заданного целого положительного числа

m. Разработайте алгоритм нахождения периода Пизано для чисел m в диапазоне от P до Q.

Решение. Понятно, что надо строить последовательность Фибоначчи и искать остатки от деления элементов последовательности на m. Последний элемент в периоде равен 0, но не каждое значение 0 заканчивает период. Поэтому надо найти 0, которые заканчивает второй период. Т.е. если остаток от деления F_k на m (k должно быть нечётным) равен 0, то надо убедиться, что для всех i от 0 до $k / 2$ выполняется $F_i \bmod m = F_{i+k/2+1} \bmod m$ (используем целочисленное деление).

Проблема в том, как определить количество чисел Фибоначчи, которых будет достаточно для решения задачи. В принципе, известно, что длина периода Пизано для некоторого числа m не больше, чем $6 \cdot m$. Поэтому можно построить последовательность из $12 \cdot Q$ чисел Фибоначчи и несколько раз пройти по массиву, ища период Пизано для всех m в диапазоне от P до Q. Но если мы этого не знаем, то надо просто в цикле строить последовательность чисел Фибоначчи, пока не будет найден период Пизано. Можно строить эту последовательность несколько раз для каждого значения m, а можно для каждого нового числа Фибоначчи находить остаток от его деления на все m в диапазоне от P до Q и продолжать цикл, пока мы не найдём период Пизано для всех m.

```
алг ПериодПизано()
нач
  цел p, q, f[100000], m, pp, k, i
  лог fpp

  ввод p, q
  если p <= 0 или q <= 0 или p > q то
    вывод "Некорректные исходные данные"
  иначе

    для m от p до q
    нц
      f[0] = 1
      f[1] = 1
      k = 1
      pp = 0                      // Период Пизано не найден
      пока pp = 0
      нц
        если f[k] mod m = 0 и k mod 2 = 1 то
          fpp = истина
          i = 0
          пока i <= k div 2 и fpp
          нц
            если f[i] mod m <> f[i + k div 2 + 1] mod m то
              fpp = ложь
            всё
            i = i + 1
          кц
          если fpp то
            pp = k div 2 + 1
          всё
        всё
        k = k + 1
        f[k] = f[k – 1] + f[k – 2]
      кц
      вывод "Период Пизано для числа ", m, " равен ", pp
    кц

  всё
кон
```