**1.** Алгоритмизация, циклы

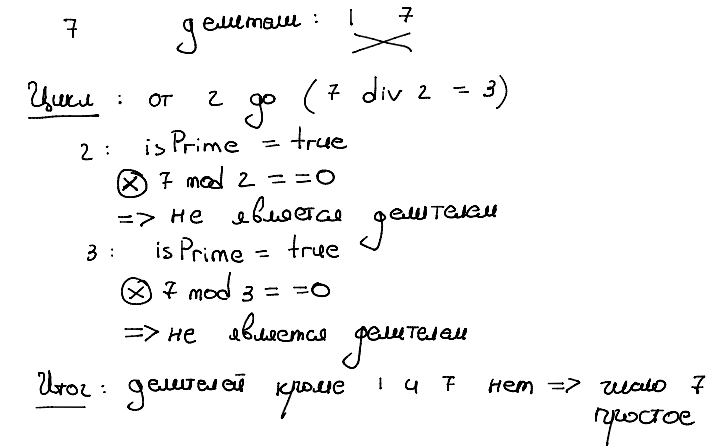
1. *Напишите программу для поиска простых чисел в заданном диапазоне. Диапазон вводится с клавиатуры.*

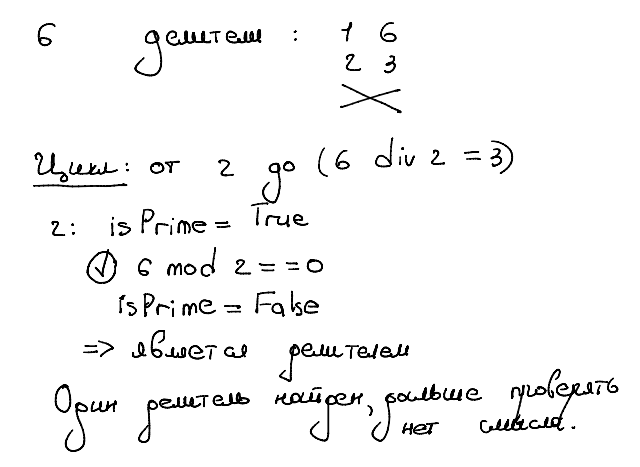
Простое число — это натуральное число больше 1, которое не является произведением двух меньших натуральных чисел. Простое число имеет ровно два делителя: 1 и само число.

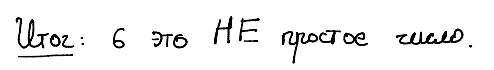
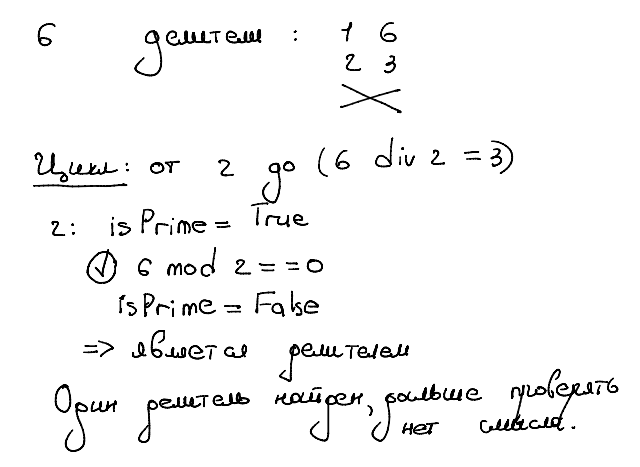
Например, 7 — простое число, потому что оно имеет только два делителя 1 и 7, а 14 — не простое число, потому что делится на 1, 2, 7 и 14.

Сформируем алгоритм, который позволит определить является ли число простым.

1. Пропускаем все отрицательные числа, 0 и 1. Число 1 — не является ни простым, ни составным числом, так как у него только один делитель — 1.
2. Вводим логическую переменную и инициализируем ее значением **True** (изначально предполагаем, что число является простым).
3. Формируем цикл, который будет двигаться от 2 до половины проверяемого числа с шагом в 1 и проверяем, делится ли наше первоначальное число на число из цикла без остатка. Если это условие выполнилось хотя бы один раз => делитель найден и проверяемое число **НЕ** простое.







Решение



***Пример вывода***:



1. *Найдите факториал числа, введенного с клавиатуры.*

Факториал числа - результат последовательного умножения числа на все натуральные числа меньшие данного числа и большие единицы. Обозначается факториал восклицательным знаком после числа — «n!».

Факториал натурального числа n можно также определить как рекуррентную функцию F (n). Определяется она следующим образом: ; .

Приведем пример нескольких начальных членов последовательности.

Решение



***Пример вывода***:



1. *Определите НОД (наибольший общий делитель) для двух чисел, введенных с клавиатуры.*

Наибольшим общим делителем двух чисел a и b называется наибольшее число, на которое a и b делятся без остатка. Для записи может использоваться аббревиатура НОД. Для двух чисел можно записать вот так: НОД (a, b). Например, для 4 и 16 НОД будет 4.

Разберем наиболее простой алгоритм решения этой задачи.

1. Задаем цикл, который начинается от 1 и идет до наибольшего из введенных чисел с шагом в 1.
2. Проверяем, если первое и второе число без остатка делится на счетчик, значит счетчик является делителем как для первого, так и для второго числа. Запоминаем его.
3. Последнее число, которое мы запомнили, будет ответом на задачу.

Перейдем к чуть более интересному варианту. Алгоритм Евклида – это алгоритм нахождения наибольшего общего делителя (НОД) пары целых чисел. Сам алгоритм при использовании деления можно представить следующим образом:

1. Большее число делим на меньшее.
2. Если делится без остатка, то меньшее число и есть НОД (следует выйти из цикла).
3. Если есть остаток, то большее число заменяем на остаток от деления.
4. Возвращаемся к 1 пункту.

Решение

1. Простое решение

2. Алгоритм Евклида

***Пример вывода***:

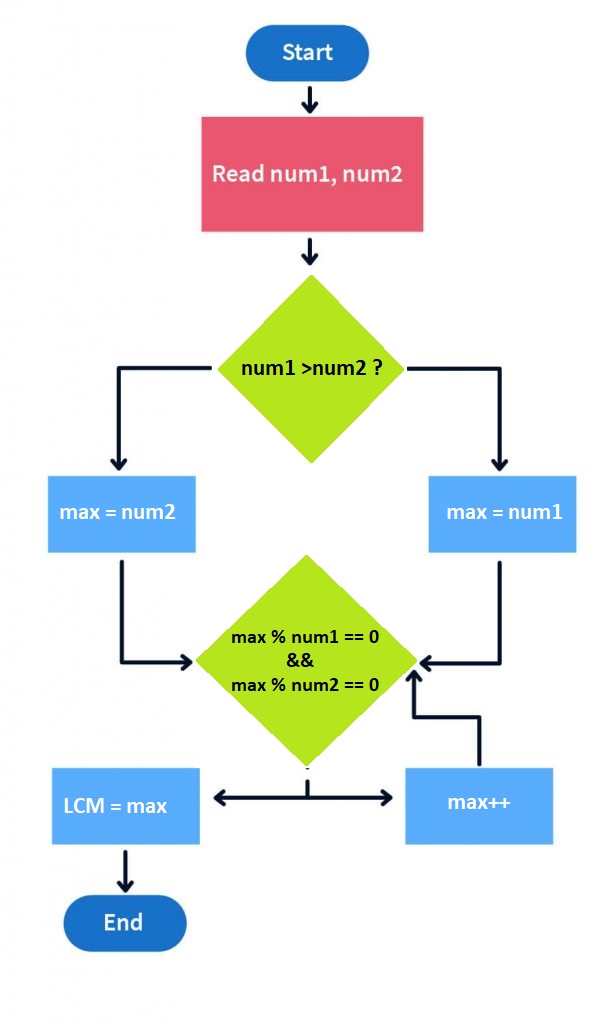


1. *Определить НОК (наименьшее общее кратное) для двух чисел. Числа вводятся с клавиатуры.*

В первом варианте решения воспользуемся наиболее примитивным способом. Введем переменную, которая будет равна максимальному из двух чисел и будем увеличивать ее на 1 до тех пор, пока не найдем число, которое без остатка делится на два наших введенных числа.

Попробуем повысить эффективность первого способа. Один из методов нахождения НОК – перемножить между собой два числа и разделить их на их НОД, который мы находили в пред. задаче.

Решение

1.  

2. 

***Пример вывода***:

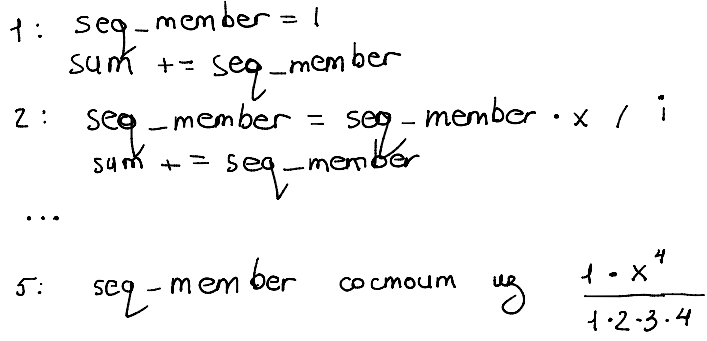


1. *Посчитайте и выведите на экран сумму ряда []. Значение переменной x и переменной n, которая отвечает за количество повторений вводятся с клавиатуры.*

Из условия можно понять, что у нас есть некая последовательность, каждый новый член которой считается по определенному правилу. В общем виде член ряда под номером можно представить следующим образом: , *где* .

Нам нужно найти сумму определенного кол-во членов этого ряда, причем наш подсчет каждый раз начинается с его первого члена.

1. Введем накопительную переменную. Не забываем ее обнулить.
2. Введем переменную, в которой будет храниться член ряда и присвоим ей значение 1, потому что первый член ряда всегда будет равен 1.
3. Запустим цикл от 1 до n c шагом один, в котором будем:
   1. Считать новый член ряда, умножив его на x, и, разделив на номер члена в последовательности.



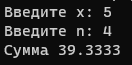
* 1. Считать общую сумму, прибавив член ряда к накопительной переменной.

1. Выведем значение накопительной переменной на экран.

Решение

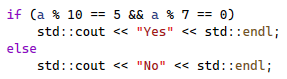


***Пример вывода***:

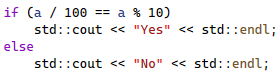


**2.** Mod и Div

1. *Найдите сумму цифр в введенном числе.*
2. *Eсли введенное с клавиатуры число оканчивается на 5 и делится на 7, то вывести «YES» иначе «NO».*



1. *Дано трехзначное число. Выяснить, является ли оно палиндромом («перевертышем»), т.е. таким числом, десятичная запись которого читается одинаково слева направо и справа налево. Усл. вариант: проверить, что на входе у нас именно трехзначное число.*



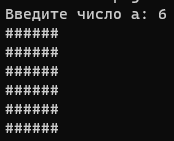
**3.** Вывод данных, циклы

1. *Напишите программу для вывода в консоли квадрата, составленного из символа #. Число a, определяющее сторону квадрата вводится с клавиатуры.*

Решение



***Пример вывода***:



1. *(Дом. Задание) Напишите программу для вывода в консоли узора в виде «алмаза», составленного из знака звездочки (\*). С клавиатуры вводится число, которые обозначает количество строк до середины «алмаза».* ***Доп. задание****: выводить на экран несколько фигур в ряд, в соответствии с тем числом, которое было введено с клавиатуры.*

***Пример вывода***:

