Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Основы алгоритмизации и программирования (ОАиП)

ОТЧЕТ

по лабораторной работе №2

Тема работы: Нахождение простых множителей

Выполнил

студент: гр. 251003 Дедов Н.Ю.

Проверил: Фадеева Е.П.

Минск 2022

Содержание

[1 Постановка задачи 3](#_Toc119950722)

[1.1 Условие задачи 3](#_Toc119950723)

[1.2 Ввод данных 3](#_Toc119950724)

[1.3 Вывод данных 3](#_Toc119950725)

[2 Структура Данных 4](#_Toc119950726)

[3 Текстовый алгоритм данных 5](#_Toc119950727)

[4 Методика решения 6](#_Toc119950728)

[4.1 Условия для ввода данных 6](#_Toc119950729)

[4.2 Проверка введенных данных 6](#_Toc119950730)

[4.3 Алгоритм решения 6](#_Toc119950731)

[5 Схема алгоритма решения задачи по ГОСТ 19.701-90 7](#_Toc119950732)

[6 Результаты расчетов 8](#_Toc119950733)

[Приложение А 9](#_Toc119950734)

[Приложение Б 11](#_Toc119950735)

# Постановка задачи

## Условие задачи

Разработать алгоритм и программу поиска для целого числа M всех простых множителей(PrimeDivisor) в порядке их возрастания. Одинаковые множители печатать столько раз, сколько раз они встречаются.

## Ввод данных

Пользователь вводит число M. Нам необходимо посчитать его простые множители (PrimeDivisor).

## Вывод данных

Вывод данных происходит в следующем виде:

Для числа M простыми множителями являются

PrimeDivisor1 = значение

PrimeDivisor2 = значение

…

Если простых делителей нет, то вывод должен выглядеть следующим образом:

Простых множителей у числа M нет.

Если введенное число M не соответствует условию ([см. условия ввода данных](#_Условия_для_ввода)), выводится:

Некорректные данные, перезапустите программу.

# Структура Данных

Таблица 1 – Данные

|  |  |  |
| --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение |
| M | Integer | Число M, вводится пользователем |
| PrimeDivisor | Integer | Простой множитель числа M |

# Текстовый алгоритм данных

Таблица 2 – Алгоритм решения

|  |  |
| --- | --- |
| Номер  Шага | Назначение шага |
|  |  |
|  | Ввод M |
|  | PrimeDivisor := 2[[1]](#footnote-1). Устанавливаем значение 2, т.к. это первое простое число |
|  | Проверка выполнение условия: (M > 1). Если условие истинно, то идти к шагу 5, иначе – к шагу 11 |
|  | Начало цикла А1. Проверка выполнение условия: (M > 1). Если условие истинно, то идти к шагу 6, иначе – к шагу 14 |
|  | Проверка выполнение условия: (M mod PrimeDivisor <> 0). Если условие истинно, то идти к шагу 7, иначе – к шагу 8 |
|  | PrimeDivisor := PrimeDivisor + 1. Идти к шагу 10 |
|  | Вывод PrimeDivisor |
|  | M := M div PrimeDivisor |
|  | Конец цикла А1. Вернуться к шагу 5 |
|  | Проверка выполнения условия: (M = 1). Если условие истинно, то идти к шагу 12, иначе – к шагу 13 |
|  | Вывод сообщения об отсутствии простых множителей. Идти к шагу 14 |
|  | Вывод сообщения о вводе некорректных данных |
|  | Останов. |

# Методика решения

## Условия для ввода данных

1. Число M должно быть целым.
2. Число M должно быть положительным.

## Проверка введенных данных

Так как пользователь может ввести неверные данные, то программа должна это отслеживать и оповестить пользователя об этом. Мы будем использовать проверку соблюдения правил ввода ([см. условия для ввода данных](#_Условия_для_ввода)). В программе это выглядит так:

If (M > 1) then

Begin

…

If (M = 1) then

Writeln ('Простых множителей у числа M нет');

Else

Writeln ('Некорректные данные, перезапустите программу.');

## Алгоритм решения

Чтобы найти все простые множители числа M, мы будем проверять его делимость на PrimeDivisor начиная с 2, т.к. это первое простое число. Если число M делится на PrimeDivisor мы выписываем PrimeDivisor, если же нет, то мы увеличиваем значение PrimeDivisor на 1. В самом начале мы проверяем больше ли число M, введенное пользователем, единицы или нет. Если оно больше, то программа ищет простые множители числа M, если же нет, то уведомляем пользователя.

# Схема алгоритма решения задачи по ГОСТ 19.701-90



Рисунок 1 – Схема алгоритма решения задачи по ГОСТ 19.701-90

# Результаты расчетов

Как результат выполнения программы, на экран выводятся следующие результаты расчетов:

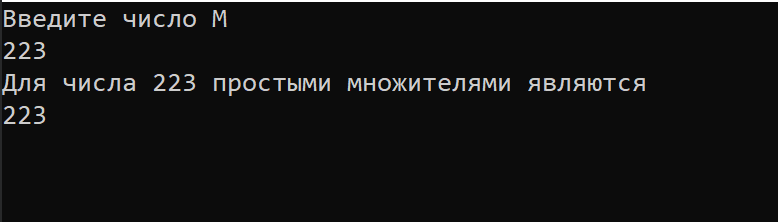


Рисунок 2 – Результаты расчетов

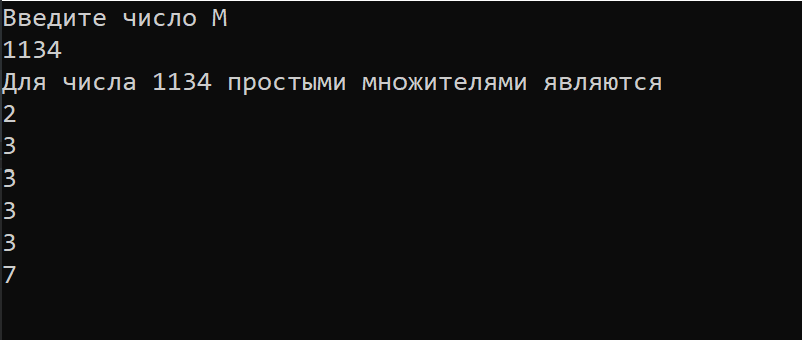


Рисунок 3 – Результаты расчетов

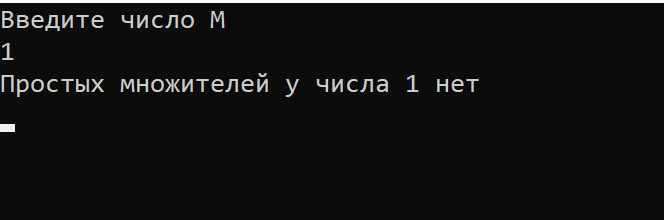


Рисунок 4 – Результаты расчетов при вводе 1

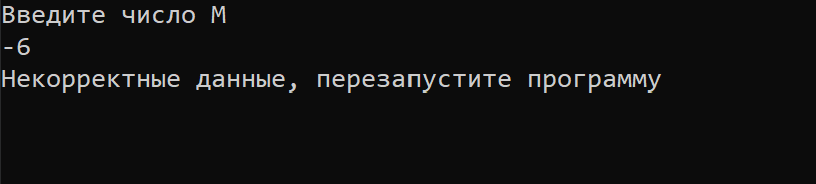


Рисунок 5 – Некорректный ввод данных

Приложение А

(обязательное)

Исходный код программы

Program Lab2;

{

For a given number M, find its’ prime divisors,

If they repeat, show all of them

}

// Use app

{$APPTYPE CONSOLE}

// Declare vars

Var

// M – number that user input

// PrimeDivisor – prime divisor

M, PrimeDivisor: integer;

Begin

// Asking a user to give us a number

Writeln ('Введите число M');

Readln (M);

// Making sure that there any divisors

If (M > 1) then

Begin

// Setting Prime\_divisor to the first prime number

PrimeDivisor := 2;

Writeln ('Для числа ', M, ' простыми множителями

являются ');

// Setting our cycle, we will find prime divisors

While (M > 1) do

Begin

// We will find residue

If (M mod PrimeDivisor <> 0) then

PrimeDivisor := PrimeDivisor + 1

Else

// If there is no residue, we will right it

Begin

Writeln (PrimeDivisor);

M := M div PrimeDivisor;

End;

End;

End

Else

If (M = 1) then

Writeln ('Простых множителей у числа M нет');

Else

Writeln ('Некорректные данные, перезапустите программу.');

Readln;

End.

Приложение Б

(обязательное)

Тестовые наборы

|  |  |  |
| --- | --- | --- |
| Номер теста | Исходные данные | Полученный результат |
| 1 | M присвоить 223 | [Рисунок 2 – Результаты расчётов](#_Результаты_расчетов) |
| 2 | M присвоить 1134 | [Рисунок 3 – Результаты расчётов](#_Результаты_расчетов) |
| 3 | M присвоить 1 | [Рисунок 4 – Результаты расчётов](#_Результаты_расчетов) |
| 4 | M присвоить -6 | [Рисунок 5 – Результаты расчётов](#_Результаты_расчетов) |

1. sTYUIOPIYGHIOPLJB [↑](#footnote-ref-1)