Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

**Лабораторна робота №** 1

по дисципліні «Теорія алгоритмів»

на тему: Вступ до теорії алгоритмів

|  |  |
| --- | --- |
| Виконав:  студент групи ІА-з91  Москаленко В.В.  Дата здачі \_\_\_\_\_\_\_\_\_\_\_\_  Захищено з балом \_\_\_\_\_\_\_\_\_\_\_ | Перевірив:  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Київ 2020

**0.Завдання.**

Написати програму, яка розкладає задане користувачем число на прості множники. Вивести на екран всі множники числа. Примітка – множники мають бути простими числами(числами у яких нема множників), див додаток 1.  
*х* є {1,10000}  
На приклад: 324 = 2\*2\*3\*3\*3\*3, 234 = 2·3·3·13

1. **Постановка проблеми.**

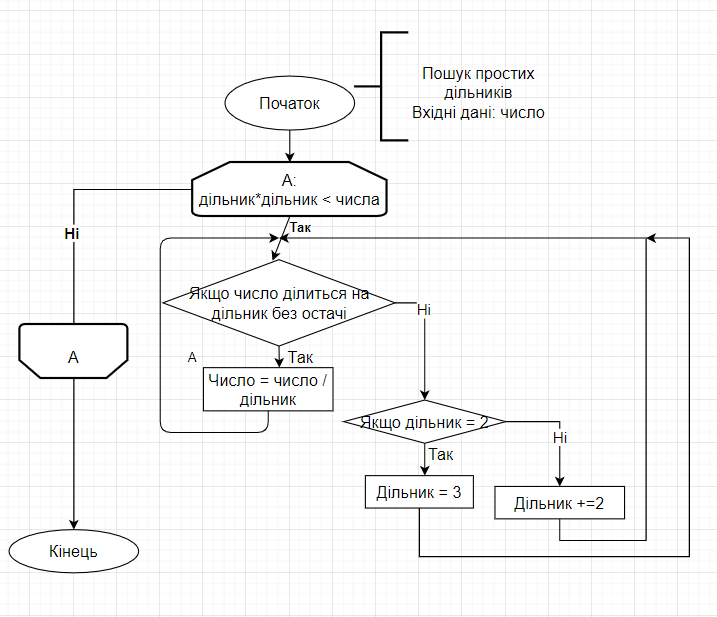
Нам відомо, що треба скласти програми, яка має розкладати задане число користувачем на прості множними. Треба визначити ці прості множники, їхню кількість і записати в аналітичній формі. Розв’язок до даного завдання існує, але не є однозначним. Так як введене число користувача може бути комплексним, або дійсним(може мати дробну частину). Конфліктів між відомими даними не має.

1. **Побудова моделі.**

Для даної задачі існує готова модель розв’язку. Будемо використовувати цикл, який буде перебирати множники і вкладений іф елс блок, який буде їх перевіряти. Даний алгоритм є широко поширений і мінімізує можливі поширені помилки.

1. **Розроблення алгоритму.**

В даному випадку будемо використовувати цикл while в якому буде порівнюватись множники з числом, а також всередині нього вкладений if...else блок в якому будемо перевіряти умови чи число ділиться на цей множник або інкрементувати цикл вайл.



1. **Перевірка правильності алгоритму.**

Даний алгоритм сканує ведене користувачем число. За допомогою метода NextInt{}скануємо число у натуральній формі, тобто перевіряємо чи вхідні дані є коректними. Даний цикл не є зацикленим, так як він всередині інкрементується і перевіряє чи даний простий множник в квадраті менше або дорівнює заданого користувачем числа. Всередині циклу ми спочатку перевіряємо чи дане число ділиться на множник без остачі або ж воно дорівнює 2 або ж ми інкрементуємо цикл на дві ітерації. На кожному кроці наш цикл буде або ітеруватися на 2 або інкрементуватися на 2 ітерації або ми будемо отримувати простий множник, окрім базового випадку це і є число 2. Даний цикл буде давати простий множник навіть на одному вхідному значенні. Якщо число є більшим 1, то алгоритм спочатку буде перевіряти ділення на 2, потім за потреби на 3 а потім інкрементуватись на дві ітерації.

1. **Реалізація алгоритму.**

|  |  |
| --- | --- |
| import java.util.Scanner; |  |
|  | |
| public class Lab1 { | |
| /\*\* | |
| \* Entry point. Tests the method {@code simpleDivision(num)} | |
| \* | |
| \* @param num parameter for {@code simpleDivision(num)}. | |
| \*/ | |
| public static void main(String[] args) { | |
|  | |
| Scanner scan = new Scanner(System.in); | |
| System.out.print("Please enter the number: "); | |
| int num = scan.nextInt(); | |
| scan.close(); | |
|  | |
| System.out.print(num + " = "); | |
| simpleDivision(num); | |
| } | |
|  | |
| /\*\* | |
| \* Returns the math calculated result | |
| \* | |
| \* @params number the entire parameter. | |
| \* @return void | |
| \*/ | |
| static void simpleDivision(int number) { | |
| int divisor = 2; | |
|  | |
| while (divisor\*divisor <= number) | |
| { | |
| if (0 == number % divisor) | |
| { | |
| System.out.print(divisor); | |
| number = number / divisor; | |
| System.out.print('\*'); | |
| } | |
| else if (2 == divisor) | |
| { | |
| divisor = 3; | |
| } | |
| else | |
| { | |
| divisor += 2; | |
| } | |
| } | |
|  | |
| System.out.println(number); | |
| } | |
|  | |
| } | |

1. **Перевірка програми.**

Програма працює як на натуральних числах так і викидає exception у випадку неправильних даних.

**7. Результат виводу програми.**

