Лабораторная работа №4

Дисциплина: Математические основы защиты информации и информационной безопасности

Аветисян Давид Артурович

Содержание

1 Цель работы **5**

2 Задание **6**

3 Выполнение лабораторной работы **7**

4 Выводы **12**

List of Tables

List of Figures

3.1 Алгоритм Евклида на языке Julia 7
3.2 Алгоритм Евклида на языке Julia 7
3.3 Бинарный алгоритм Евклида на языке Julia 8
3.4 Бинарный алгоритм Евклида на языке Julia 8
3.5 Расширенный алгоритм Евклида на языке Julia 9
3.6 Расширенный алгоритм Евклида на языке Julia 9
3.7 Расширенный бинарный алгоритм Евклида на языке Julia 10
3.8 Расширенный бинарный алгоритм Евклида на языке Julia 10

1 Цель работы

Реализовать алгоритмы вычисления наибольшего общего делителя.

2 Задание

1. Реализовать алгоритм Евклида.
2. Реализовать бинарный алгоритм Евклида.
3. Реализовать расширенный алгоритм Евклида.
4. Реализовать расширенный бинарный алгоритм Евклида.

3 Выполнение лабораторной работы

Данная работа была выполнена на языку Julia.

1) Для реализации алгоритма Евклида была написана следующая программа.

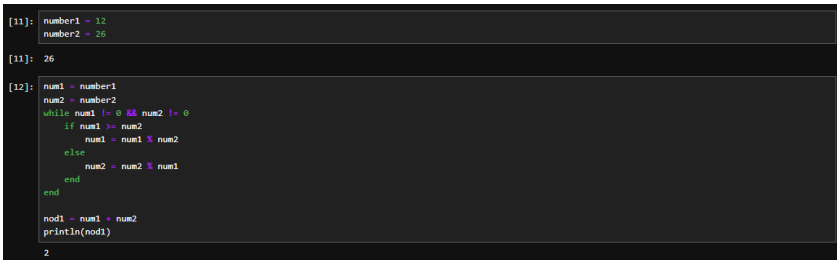

Figure 3.1: Алгоритм Евклида на языке Julia

В данной программе: - 1-5 строки: задание чисел, НОД которых ищем. - 6-12 строки: реализация самого алгоритма Евклида: делим наибольшее число на наименьшее и записываем остаток до тех пор, пока одно из них не обнулится. - 14-15 строки: записываем НОД в переменную и выводим.

Мы можем видеть результат для двух случае на рисунках выше и ниже. Программа работает верно.

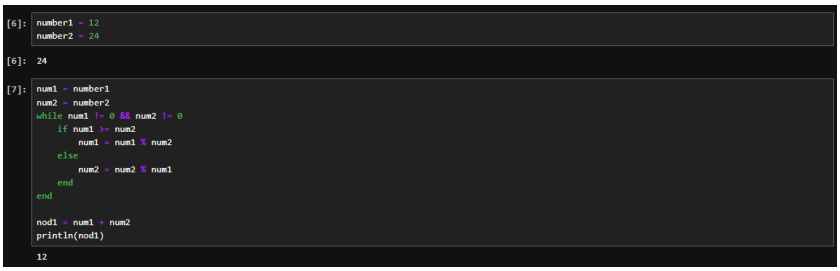

Figure 3.2: Алгоритм Евклида на языке Julia

2) Для реализации бинарного алгоритма Евклида была написана следующая программа.

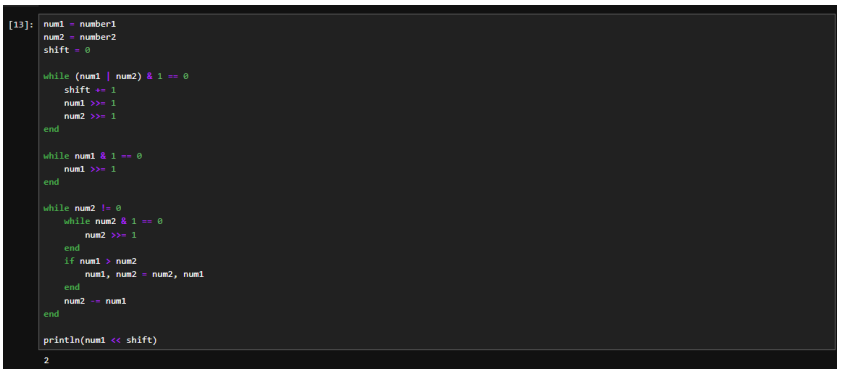

Figure 3.3: Бинарный алгоритм Евклида на языке Julia

В данной программе: - 1-3 строки: задание чисел, НОД которых ищем, и обнуление “сдвига”. - 5-23 строки: реализация самого бинарного алгоритма Евклида: смотрим на четность получающихся значений и записываем, насколько нам небходимо “сдвинуть” число, чтобы получить НОД. - 25 строка: сдвиг влево и вывод получившегося НОД.

Мы можем видеть результат для двух случае на рисунках выше и ниже. Программа работает верно.

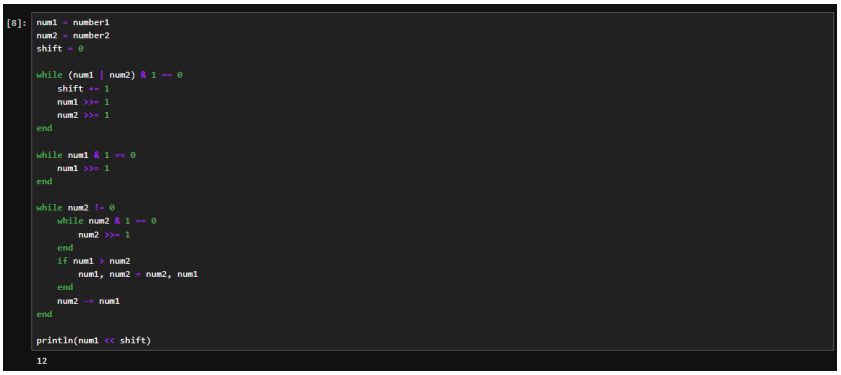

Figure 3.4: Бинарный алгоритм Евклида на языке Julia

3) Для реализации расширенного алгоритма Евклида была написана следующая программа.

Figure 3.5: Расширенный алгоритм Евклида на языке Julia

В данной программе: - 1 строка: задание рекурсивной функции. - 2-3 строки: если второе число равно нулю, возвращаем ответ из трёх чисел. - 5-7 строки: в ином случае запускаем рекурсию, а затем выводим ответ согласно формуле на строке 7. - 8 строка: возвращаем вывод в качестве получившегося НОД; числа, что нужно домножить на первую цифру и на вторую, чтобы получить НОД. - 11 строка: вызов функции и сохранение данных в переменные. - 12 строка: вывод на экран.

Мы можем видеть результат для двух случае на рисунках выше и ниже. Программа работает верно.

Figure 3.6: Расширенный алгоритм Евклида на языке Julia

4) Для реализации расширенного бинарного алгоритма Евклида была написана следующая программа.

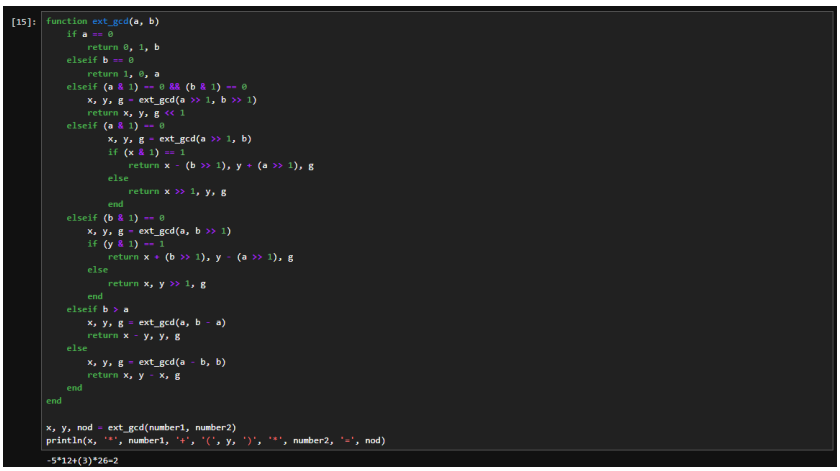

Figure 3.7: Расширенный бинарный алгоритм Евклида на языке Julia

Данная программа работает рекурсивно, рассматривая 4 случая: 1. a четное 2. a нечетное и b четное 3. a нечетное и b нечетное, b > a 4. a нечетное и b нечетное, a > b

Каждая рекурсия сдвигает биты в цифрах, формирую окончательный ответ. В итоге выводит три значения: НОД; числа, которые нужно домножить на первую цифру и на вторую, чтобы получить НОД.

Мы можем видеть результат для двух случае на рисунках выше и ниже. Программа работает верно.

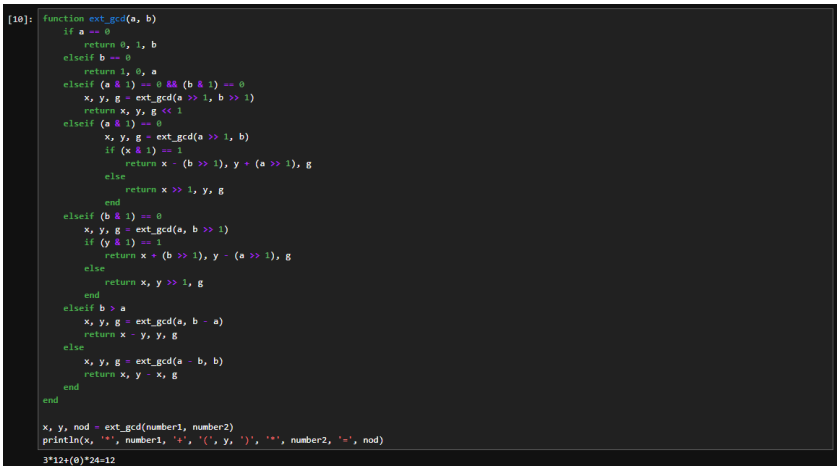

Figure 3.8: Расширенный бинарный алгоритм Евклида на языке Julia

Необходимо обратить внимание, что расширенные алгоритмы выводят разные

множители, однако оба ответа верны и дают верный НОД.

4 Выводы

Я реализовал алгоритмы вычисления наибольшего общего делителя.