Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Дисциплина «Основы защиты информации»

Отчёт по практическому занятию №6

Студент:Шатерник Г.И.

ФИТ 2 курс 3 группа

Преподаватель: Ржеутская Н.В.

Минск 2023 г.

**Практическое занятие №6**

**«Криптографическая защита информации»**

Цель: получение основных сведений из курса теории чисел.

**Теоретическое введение**

Ниже рассматриваются: *N* – множество натуральных чисел, *Z* – множество рациональных чисел. Множество целых чисел *Z* – счетное, состоит из элементов 0; ±1; ±2; …; ± *n*,…. На нем определены две алгебраические операции – сложение и умножение. Эти операции обладают следующими свойствами (для любых ):

1. ассоциативность: ; ;

2. коммутативность: ; ;

3. существует нейтральный элемент – 0 и 1 соответственно:



4.  – закон дистрибутивности;

5. для каждого целого  существует единственное противоположное, то есть такое целое *b*, что *a* + *b* = *b* + *a* = 0.

*Теорема 2.1* (*О делении с остатком*). Для любых целых чисел *a* и *b*, , существует единственные целые числа *q* и  , такие, что .

В этом равенстве  называют остатком, а  – частным (неполным частным – при ) от деления *a*  на  При *r* = 0 величины *b* и *q* называют делителями или множителями числа *а*. Читатель со школьной скамьи умеет находить частное и остаток методом деления уголком.

*Следствие.* Пусть  – натуральное число,  Для всякого целого числа *a*  и максимального целого  с условием  существуют единственные целые  такие, что 

Такое равенство записывают сокращённо  или  (если *b* известно по контексту) и называют записью числа *a* в *b* – ичной позиционной системе счисления или системе счисления по основанию *b*. Нам кажется естественной привычная десятичная позиционная система записи целых чисел . В различных ситуациях более удобными оказываются другие основания. К примеру, во всех компьютерах на микроуровне вычисления проводятся в двоичной системе счисления. Для перехода к ней с десятичной применяют промежуточную – 16 - ричную систему счисления.

*Лемма 2.1.* Если в равенстве  все слагаемые – целые числа и все, кроме может быть одного, делятся на целое , то и это исключенное слагаемое делится на .

**Определение 2.1*.***Если целые числа  делятся на целое , то *d*  называют их *общим делителем*.

В дальнейшем речь идет только о положительных целых делителях.

**Определение 2.2.** Максимальный из общих делителей целых чисел  называется их *наибольшим общим делителем* и обозначается через НОД ().

*Теорема 2.2.* Если *,* то НОД *(a, b)*=НОД *(b, c).*

Теорема 2.2 позволила Евклиду (примерно 2300 лет тому назад) обосновать следующий факт.

*Теорема 2.3.* Наибольший общий делитель целых чисел  *a* и *b*   равен последнему отличному от нуля остатку цепочки равенств:

*;*

*;*

*…………………*

**

**

то есть  *=* НОД *.*

Теорема 2.3 формулирует алгоритм Евклида нахождения наибольшего общего делителя целых чисел. Его вариантом является следующий – второй способ вычисления наибольшего общего делителя по алгоритму Евклида – вычисляем последовательно разности  до получения последней ненулевой разности, которая и совпадает с НОД *(a, b).*

Основные свойства сравнений:

**1.** Пусть *.* Тогда  для всякого целого *c*, то есть к обеим частям сравнения можно добавить (или вычесть из обеих частей) одно и то же число.

**2.** Сравнения можно почленно складывать и вычитать: если **, *,* то  

**3.** Сравнения можно почленно перемножать: если ** *,* то **.

**4.** Сравнения можно почленно возводить в любую натуральную степень: если *,* то **.

**5.** Если в сравнении ** числа *a*, *b*, *m* имеют общий множитель *d*, то на него сравнение можно сократить: **.

**6.** Сравнение можно сократить на общий множитель, взаимно простой с модулем: если **, НОД (*d*, *m*) = 1, то из сравнения  следует сравнимость  и  по модулю .

**7.** Сравнение можно умножить на любой целый множитель: если **, то  для всякого целого *t*.

**8.** Рефлексивность: ** для любого целого *а* и всякого натурального *m* >1.

**9.** Симметричность: если **, то **.

**10.** Транзитивность: если **, **, то .

*Теорема 2.10*(*Малая теорема Ферма*). Пусть *p –* простое число и целое число *a* не делится на . Тогд*а .*

Теория сравнений и малая теорема Ферма позволяют быстро находить остаток от деления большого числа на простое число.

# **Индивидуальное задание**

*Вариант 21.*

**1.Найти канонические разложения чисел а и b.**

**а = 5336161097, b = 196210799.**

**Код**

def divide(arg):

divider=2

out\_arr=[]

while(arg!=1):

if(arg%divider ==0):

#print(divider )

out\_arr.append(divider)

arg=arg//divider

divider=2

else:

divider+=1

return out\_arr

print("число a")

inp\_var\_a=int(input())

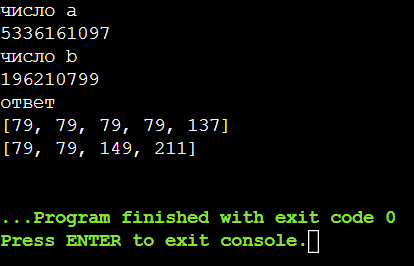
print("число b")

inp\_var\_b=int(input())

print("ответ")

print(divide(inp\_var\_a))

print(divide(inp\_var\_b))



Результат

**2. Найти НОД (5336161097,196210799) пользуясь a) алгоритмом Евклида, б) разложением чисел на простые множители.**

**Код**

def divide(arg):

divider=2

out\_arr=[]

while(arg!=1):

if(arg%divider ==0):

#print(divider )

out\_arr.append(divider)

arg=arg//divider

divider=2

else:

divider+=1

return out\_arr

def euclid(arg\_a,arg\_b):

while((arg\_a!=0) and (arg\_b!=0)):

if(arg\_a>arg\_b):

arg\_a=arg\_a%arg\_b

else:

arg\_b=arg\_b%arg\_a

return arg\_a+arg\_b

def gcd(a, b):

a\_factors = divide(a)

b\_factors = divide(b)

gcd\_value = 1

i = 0

j = 0

while i < len(a\_factors) and j < len(b\_factors):

if a\_factors[i] == b\_factors[j]:

gcd\_value \*= a\_factors[i]

i += 1

j += 1

elif a\_factors[i] < b\_factors[j]:

i += 1

else:

j += 1

return gcd\_value

print("число a")

inp\_var\_a=int(input())

print("число b")

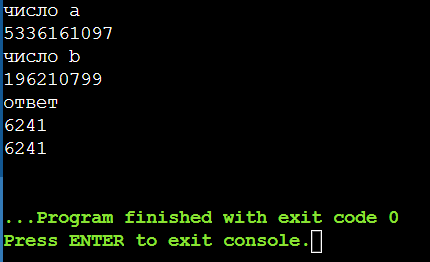
inp\_var\_b=int(input())

print("ответ")

print(gcd (inp\_var\_a,inp\_var\_b))

print(euclid(inp\_var\_a,inp\_var\_b))

Результат



**3. С помощью расширенного алгоритма Евклида найти целые u, v, удовлетворяющие соотношению Безу: au + bv = НОД .**

**Код**

def baesy(a,b):

prev\_x, x = 1, 0

prev\_y, y = 0, 1

while b != 0:

quotient = a // b

a, b = b, a % b

prev\_x, x = x, prev\_x - quotient \* x

prev\_y, y = y, prev\_y - quotient \* y

return prev\_x, prev\_y

print("число a")

inp\_var\_a=int(input())

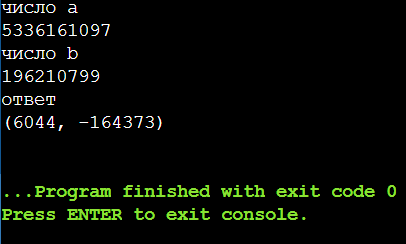
print("число b")

inp\_var\_b=int(input())

print("ответ")

print(baesy(inp\_var\_a,inp\_var\_b))

Результат



**4 Найти остаток от деления 1998 2001 на 19.Код**

def divide\_looping(mult,div):

    buff\_val=mult%div

    counter=0

    remult=mult

    while(True):

        remult\*=mult

        counter+=1

        if(buff\_val==remult%div):

            break

    return counter

def div\_select(iter,mult,div,power):

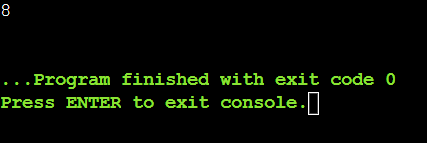
    tot\_divider=power%iter

    total\_for\_div=pow(mult,tot\_divider)

    return total\_for\_div%div

print(div\_select(divide\_looping(1998,19),1998,19,2001))

Результат



**Вывод:** в ходе работы были получены основные сведения из курса теории чисел**.**