

Отчёт по лабораторной работе 2

Студент: Кочкожаров Иван Вячеславович

Группа: M8O-308B-22

Цель работы: получить вариант на основе хеш-функции ГОСТ Р 34.11-2012 (Стрибог) и выполнить нетривиальное разложение чисел a и b на простые множители.

1 Постановка задачи

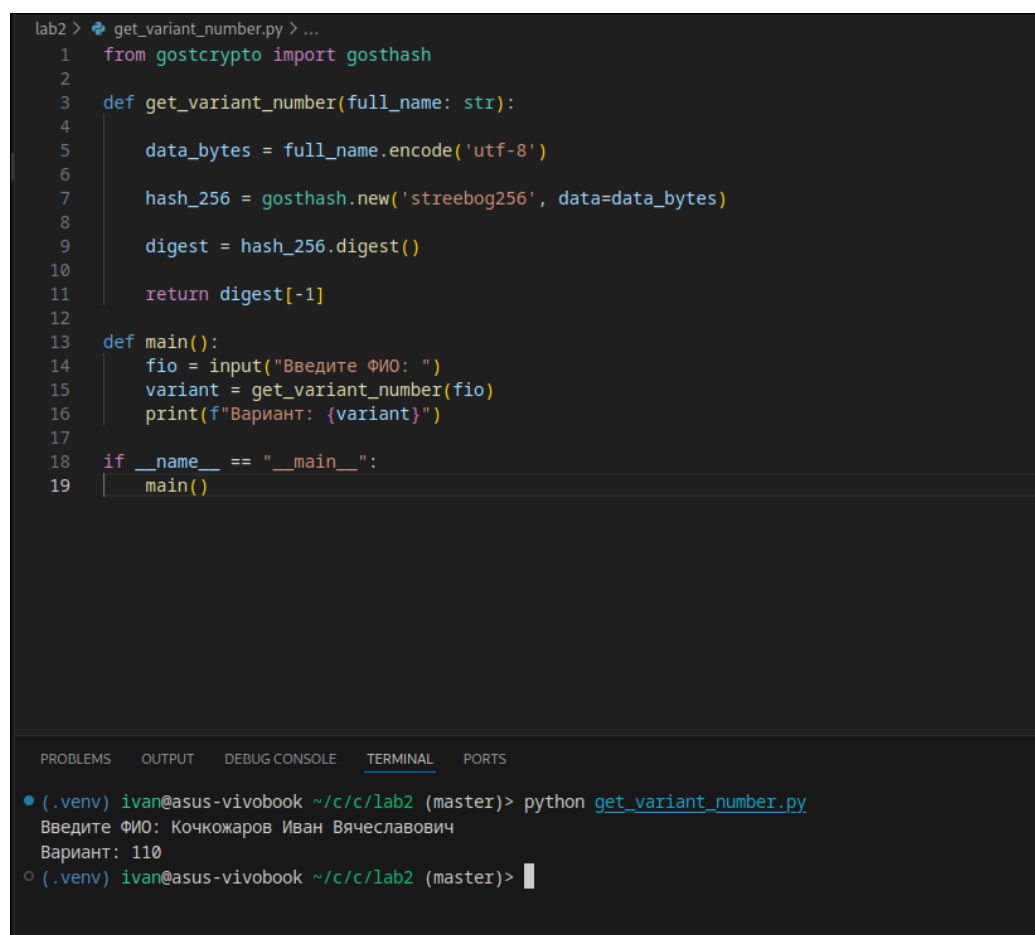
На вход хеш-функции Стрибог подаётся строка с ФИО автора. Из выходного 256-битного хеша берутся младшие 8 бит — число от 0 до 255, которое и служит номером варианта. Для данного варианта необходимо:

1. Извлечь значения a и b из списка вариантов.
2. Разложить каждое из чисел a и b на нетривиальные сомножители.

2 Получение номера варианта

```
1 from gostcrypto import gosthash
2
3 def get_variant_number(full_name: str):
4     data_bytes = full_name.encode('utf-8')
5     hash_256 = gosthash.new('streebog256', data=
        data_bytes)
6     digest = hash_256.digest()
7     return digest[-1]
```

Листинг 1: Функция для вычисления варианта по ФИО



```
lab2 > get_variant_number.py > ...
1 from gostcrypto import gosthash
2
3 def get_variant_number(full_name: str):
4
5     data_bytes = full_name.encode('utf-8')
6
7     hash_256 = gosthash.new('streebog256', data=data_bytes)
8
9     digest = hash_256.digest()
10
11     return digest[-1]
12
13 def main():
14     fio = input("Введите ФИО: ")
15     variant = get_variant_number(fio)
16     print(f"Вариант: {variant}")
17
18 if __name__ == "__main__":
19     main()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
• (.venv) ivan@asus-vivobook ~/c/c/lab2 (master)> python get_variant_number.py
Введите ФИО: Кочкожаров Иван Вячеславович
Вариант: 110
○ (.venv) ivan@asus-vivobook ~/c/c/lab2 (master)> █
```

Рис. 1: Номер варианта, полученный из ФИО

Запустив скрипт с ФИО «*Кочкожаров Иван Вячеславович*», получили:

Результат: номер варианта: 110.

$a = 96549329211410878220407731243138541789894262114809366396824543095582747523721$

$b = 32317006071311007300714876688669951960444102669715484032130345427524655138867890893197201411522913463688717960921898019494119559150490921095088153368755819279948894891352978038724364814666828075403374368623252601199573103732364197539194324809397600124011761205890421945836628563695877139401393419600972620930627469732652285701750217853272073677807696706525660813610654888881988572347249424515008349686030605536267873810768793941519437795740337005159565713224668295133294397929491874554881716226163300019068770229099948764818609237849953239277955372804162008595934215621479331566395118297110850271069157386069736822127$

3 Разбор алгоритма факторизации

3.1 Разложение числа a

Для числа a классический подход через `sympy.factorint` даёт ответ за малое время:

```
1 import re
2 from PyPDF2 import PdfReader
3 from sympy import factorint
4
5 factors_a = factorint(a)
6 print("Factor_a:")
7 print("□*□".join(str(p) for p, exp in factors_a.items()
8                 for _ in range(exp)))
```

Листинг 2: Разложение a с помощью Sympy

Результат: $a = 15381195539196285749 \times 6277101735386680763835789423207666416102355444464034513029$

3.2 «Трудное» разложение числа b

Число b из условия слишком велико для прямого перебора. Поэтому была применена следующая идея:

- Было предположено, что каждое b_i состоит ровно из двух простых сомножителей.

- Для каждого другого варианта b_j вычисляется $\gcd(b, b_j)$.
- Если $\gcd(b, b_j)$ — простое число p и b/p — простое число, то мы нашли два множителя.

Соответственно, для автоматизации перебора был написан скрипт:

```

1 from math import gcd
2 from sympy import isprime
3 #bs - all b from other variants
4 for other_b in bs:
5     d = gcd(b, other_b)
6     if isprime(d):
7         p = d
8         q = b // d
9         if isprime(q):
10             print("b_{}={}*{}".format(p, q))
11             break

```

Листинг 3: Поиск простого делителя через НОД

Предположение о числах оказалось верным, и таким образом удалось быстро найти нетривиальное разложение числа b .

Результат:

$b = 24103124269210325885801166060283141129120932479456889513596$
 $750390652573915918032006690850241073460496634487662808880047878624$
 $169787949583249696129878907746514552133393816252247707820779176814$
 $996768455431373878200575973458579045991094613871220995079649978156$
 $413423006776294733552816174284117941639677858703703689691092215919$
 $430542320115627584500805795878509009937148922834766466311815150638$
 $048733751822605062469928378987059710125258433244012329868570047606$
 $09160377 \times 1340780792994259709957402499820584612747936582059239337$
 $772356144372176403007354697680187429816690342769003185818648605085$
 $3753882811946569946433649413627751$

4 Выводы

1. Метод вычисления варианта по хеш-функции Стрибог продемонстрирован и реализован в виде скрипта на Python.
2. Число a было факторизовано стандартными средствами SymPy.

3. Для числа b найден эффективный приём через поиск общих делителей с другими вариантами, для перебора был применен парсинг pdf.
4. В отчёте приведены алгоритмы и результаты разложения.

Приложение. Исходный код

```
1 from gostcrypto import gosthash
2
3 def get_variant_number(full_name: str):
4
5     data_bytes = full_name.encode('utf-8')
6
7     hash_256 = gosthash.new('streebog256', data=
8         data_bytes)
9
10    digest = hash_256.digest()
11
12    return digest[-1]
13
14 def main():
15     fio = input("                : ")
16     variant = get_variant_number(fio)
17     print(f"                : {variant}")
18
19 if __name__ == "__main__":
20     main()
```

Листинг 4: get_variant_number.py

```
1 import re
2 from sympy import factorint, isprime
3 from math import gcd
4 from PyPDF2 import PdfReader
5 from get_variant_number import get_variant_number
6
7
8 def parse_pdf(pdf_path):
9     reader = PdfReader(pdf_path)
10    full_text = ""
11    for page in reader.pages:
12        full_text += page.extract_text()
13
```

```

14     text_clean = re.sub(r"\s+", "", full_text)
15
16     pattern = r"a\[([0-9]+\)]=([0-9]+).*?b\[([1]\)]=([0-9]+)
17         "
18
19     matches = re.findall(pattern, text_clean, re.DOTALL)
20     pairs = [(int(a), int(b)) for _, a, b in matches]
21     return pairs
22
23 def main():
24     pdf_path = "
25     full_name = "
26         "
27
28     variant_number = get_variant_number(full_name)
29     print(f"
30         :_{full_name}")
31     print(f"
32         :_{variant_number}")
33
34     pairs = parse_pdf(pdf_path)
35     a = pairs[variant_number][0]
36     b = pairs[variant_number][1]
37     print(f"a_{a}")
38     print(f"b_{b}")
39
40     bs = [b for _, b in pairs]
41     bs.pop(variant_number)
42     factors_b=[]
43     for other_b in bs:
44         divisor1 = gcd(b, other_b)
45         if (isprime(divisor1)):
46             divisor2 = b // divisor1
47             if (isprime(divisor2)):
48                 factors_b.append(divisor1)
49                 factors_b.append(divisor2)
50                 break
51
52     print("
53         _b:")
54     print(*factors_b, sep='_*_)
55
56     print("
57         _a:")
58     factors_a_dict = factorint(a)
59     factors_a = [key for key, count in factors_a_dict.
60         items() for _ in range(count)]
61     print(*factors_a, sep='_*_)
62
63 if __name__ == "__main__":

```

54

```
main()
```

Листинг 5: main.py