

Отчёт по лабораторной работе №2 по дисциплине Компьютерный практикум по статистическому анализу данных

Структуры данных

Шаповалова Диана Дмитриевна

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
2.1	Кортежи	6
2.2	Словари	7
2.3	Множества	8
2.4	Массивы	10
2.5	Задания для самостоятельного выполнения	11
2.6	1. Даны множества: $A = \{0, 3, 4, 9\}$, $B = \{1, 3, 4, 7\}$, $C = \{0, 1, 2, 4, 7, 8, 9\}$. Найти $A \cap B \cap C$	11
2.7	2. Приведите свои примеры с выполнением операций над множествами элементов разных типов	12
2.8	3. Создайте разными способами:	14
2.9	3.1) массив $(1, 2, 3, \dots, n-1, n)$, n выберите больше 20;	14
2.10	3.2) массив $(n, n-1, \dots, 2, 1)$, n выберите больше 20;	15
2.11	3.3) массив $(1, 2, 3, \dots, n-1, n, n-1, \dots, 2, 1)$, n выберите больше 20;	16
2.12	3.4) массив с именем tmp вида (4, 6, 3);	17
2.13	3.5) массив, в котором первый элемент массива tmp повторяется 10 раз;	18
2.14	5. Подключите пакет Primes (функции для вычисления простых чисел). Сгенерируйте массив myprimes, в котором будут храниться первые 168 простых чисел. Определите 89-е наименьшее простое число. Получите срез массива с 89-го до 99-го элемента включительно, содержащий наименьшие простые числа.	19
2.15	6. Вычислите следующие выражения	20
3	Выводы	21

Список иллюстраций

2.1	Выполняем примеры по Кортежам	7
2.2	Выполняем примеры по Словарям	8
2.3	Выполняем примеры по Множествам	9
2.4	задание 1	11
2.5	задание 2	12
2.6	задание 2	13
2.7	задание 3.1	14
2.8	задание 3.2	15
2.9	задание 3.3	16
2.10	задание 3.4	17
2.11	задание 3.5	18
2.12	задание 5	19
2.13	задание 6	20

Список таблиц

1 Цель работы

Основная цель работы — изучить несколько структур данных, реализованных в Julia, научиться применять их и операции над ними для решения задач.

2 Выполнение лабораторной работы

2.1 Кортежи

Примеры кортежей:

#пустой кортеж:

()

#кортеж из элементов типа String:

favoritelang = ("Python","Julia","R")

#кортеж из целых чисел:

x1 = (1, 2, 3)

#кортеж из элементов разных типов:

x2 = (1, 2.0, "tmp")

#именованный кортеж:

x3 = (a=2, b=1+2)

Примеры операций над кортежами:

#длина кортежа x2:

length(x2)

и т.д.

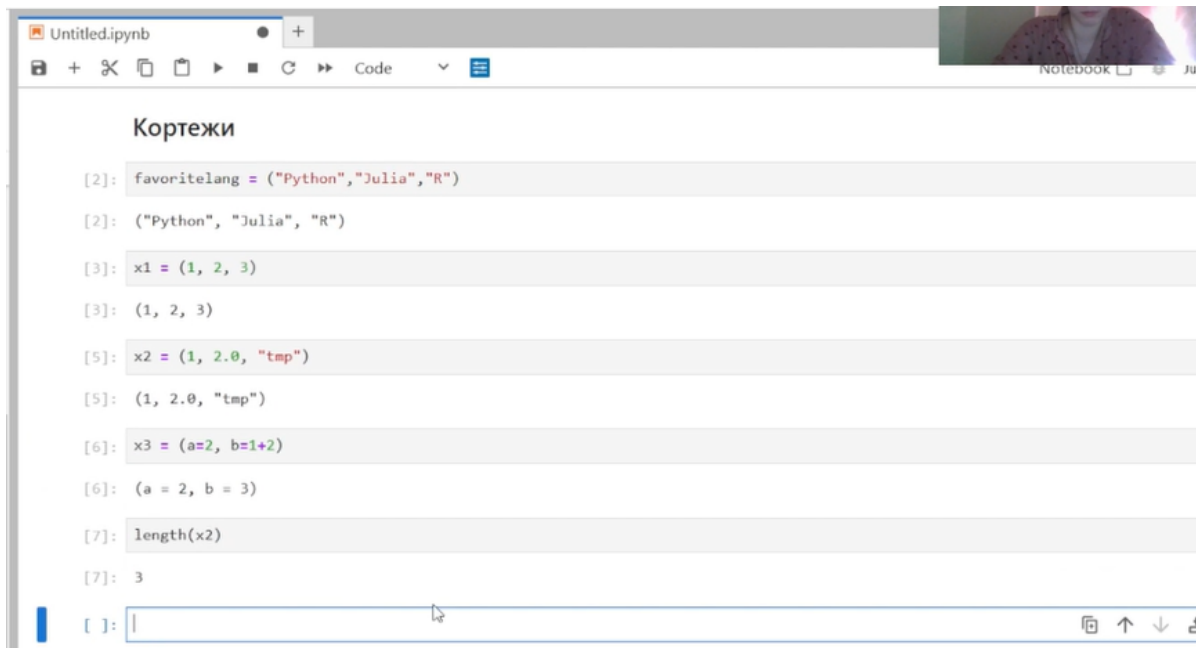


Рис. 2.1: Выполняем примеры по Кортежам

2.2 Словари

Словарь — неупорядоченный набор связанных между собой по ключу данных.

Синтаксис определения словаря:

`Dict(key1 => value1, key2 => value2, ...)`

Примеры словарей и операций над ними:

#создать словарь с именем phonebook:

`phonebook = Dict("Иванов И.И." => ("867-5309", "333-5544"), "Бухгалтерия" => "555-2368")`

#вывести ключи словаря:

`keys(phonebook)`

#вывести значения элементов словаря:

`values(phonebook)`

```

[13]: phonebook = Dict{"Иванов И.И." => ("867-5309", "333-5544"), "Бухгалтерия" => "555-2368"}

[13]: Dict{String, Any} with 2 entries:
      "Бухгалтерия" => "555-2368"
      "Иванов И.И." => ("867-5309", "333-5544")

[14]: keys(phonebook)

[14]: KeySet for a Dict{String, Any} with 2 entries. Keys:
      "Бухгалтерия"
      "Иванов И.И."

[15]: values(phonebook)

[15]: ValueIterator for a Dict{String, Any} with 2 entries. Values:
      "555-2368"
      ("867-5309", "333-5544")

[16]: pairs(phonebook)

[16]: Dict{String, Any} with 2 entries:
      "Бухгалтерия" => "555-2368"
      "Иванов И.И." => ("867-5309", "333-5544")

[17]: haskey(phonebook, "Иванов И.И.")

[17]: true

[18]: phonebook["Сидоров П.С."] = "555-3344"

[18]: "555-3344"

```

Рис. 2.2: Выполняем примеры по Словарям

2.3 Множества

Множество, как структура данных в Julia, соответствует множеству, как математическому объекту, то есть является неупорядоченной совокупностью элементов какого-либо типа. Возможные операции над множествами: объединение, пересечение, разность; принадлежность элемента множеству.

Примеры множеств и операций над ними:

#создать множество из четырёх целочисленных значений:

A = Set([1, 3, 4, 5])

#создать множество из 11 символьных значений:

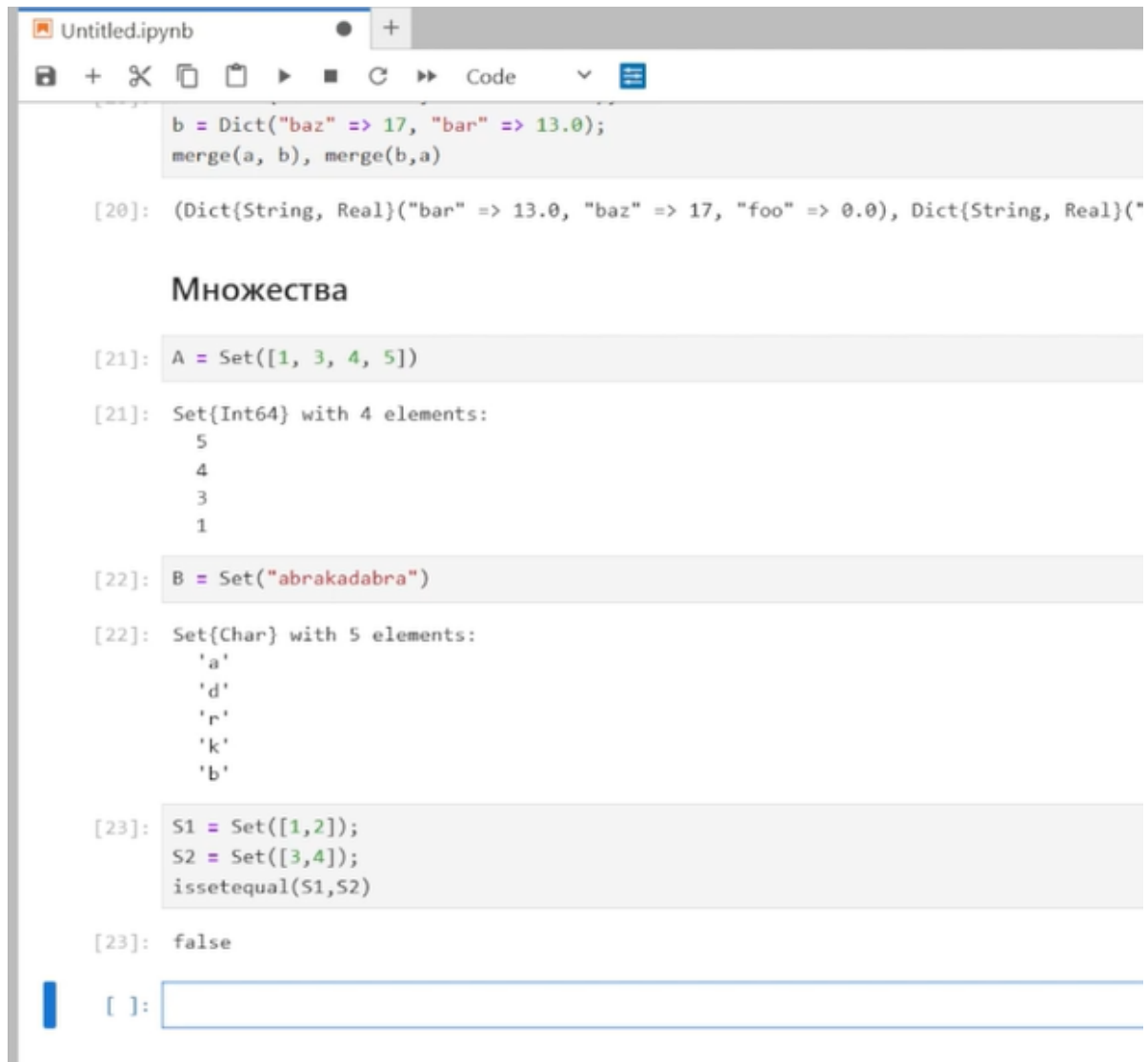

```
B = Set("abrakadabra")
```

#проверка эквивалентности двух множеств:

```
S1 = Set([1,2]);
```

```
S2 = Set([3,4]);
```

```
issetequal(S1,S2)
```



The screenshot shows a Jupyter Notebook interface with the following content:

```
Untitled.ipynb
```

Code cell [20]:

```
b = Dict{"baz" => 17, "bar" => 13.0};  
merge(a, b), merge(b,a)
```

Output [20]:

```
(Dict{String, Real}("bar" => 13.0, "baz" => 17, "foo" => 0.0), Dict{String, Real}("
```

Множества

Code cell [21]:

```
A = Set([1, 3, 4, 5])
```

Output [21]:

```
Set{Int64} with 4 elements:  
 5  
 4  
 3  
 1
```

Code cell [22]:

```
B = Set("abrakadabra")
```

Output [22]:

```
Set{Char} with 5 elements:  
'a'  
'd'  
'r'  
'k'  
'b'
```

Code cell [23]:

```
S1 = Set([1,2]);  
S2 = Set([3,4]);  
issetequal(S1,S2)
```

Output [23]:

```
false
```

Input cell []:

```
[ ]:
```

Рис. 2.3: Выполняем примеры по Множествам

2.4 Массивы

Массив — коллекция упорядоченных элементов, размещённая в многомерной сетке. Векторы и матрицы являются частными случаями массивов.

Общий синтаксис одномерных массивов:

```
array_name_1 = [element1, element2, ...]
```

```
array_name_2 = [element1 element2 ...]
```

Примеры массивов:

#создание пустого массива с абстрактным типом:

```
empty_array_1 = []
```

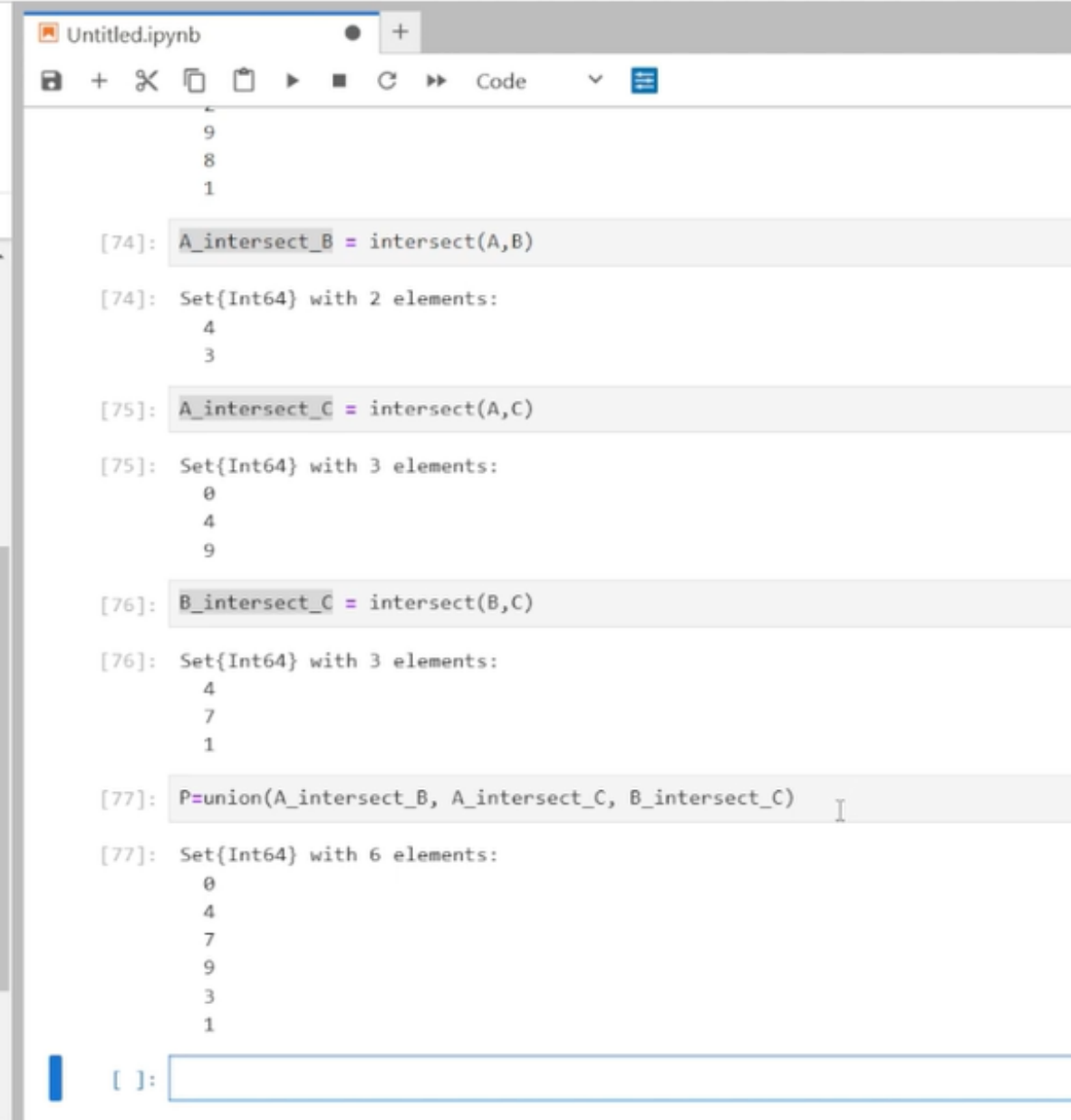
#создание пустого массива с конкретным типом:

```
empty_array_2 = (Int64)[]
```

```
empty_array_3 = (Float64)[]
```

2.5 Задания для самостоятельного выполнения

2.6 1. Даны множества: $A = \{0, 3, 4, 9\}$, $B = \{1, 3, 4, 7\}$, $C = \{0, 1, 2, 4, 7, 8, 9\}$. Найти $P = A \cap B \cup A \cap C \cup B \cap C$.



```
Untitled.ipynb
[74]: A_intersect_B = intersect(A,B)
[74]: Set{Int64} with 2 elements:
      4
      3

[75]: A_intersect_C = intersect(A,C)
[75]: Set{Int64} with 3 elements:
      0
      4
      9

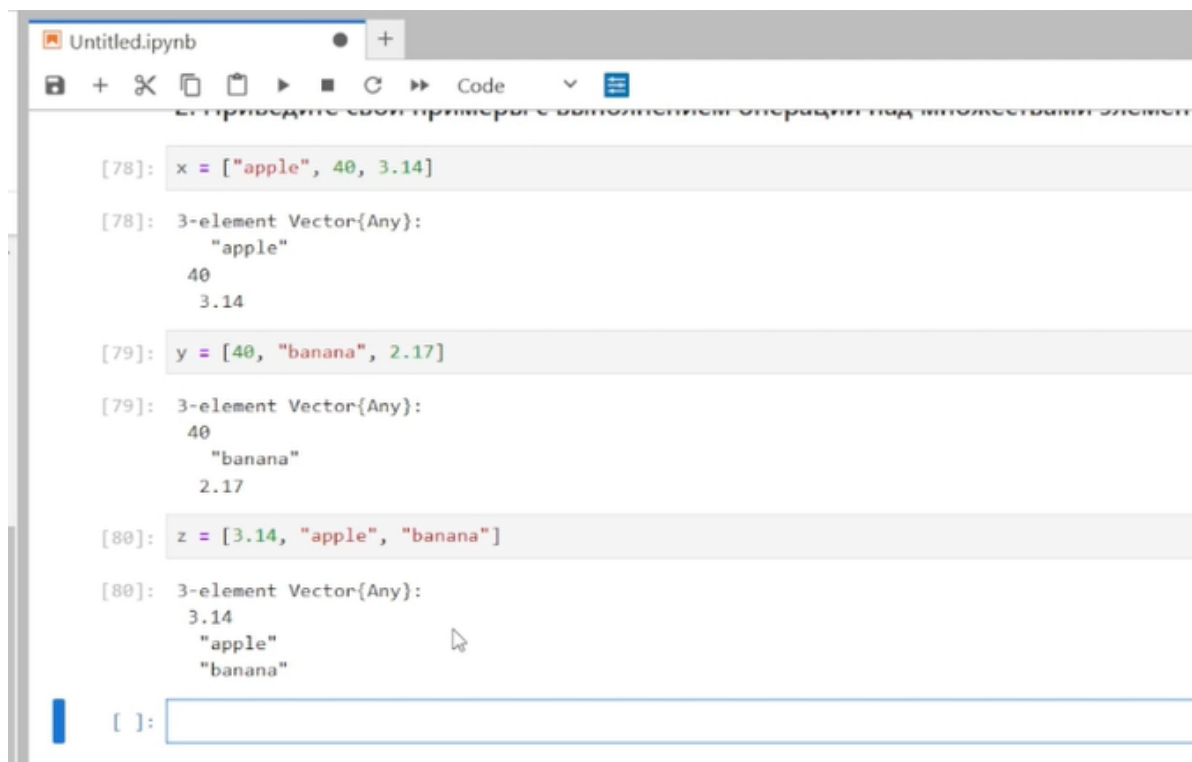
[76]: B_intersect_C = intersect(B,C)
[76]: Set{Int64} with 3 elements:
      4
      7
      1

[77]: P=union(A_intersect_B, A_intersect_C, B_intersect_C)
[77]: Set{Int64} with 6 elements:
      0
      4
      7
      9
      3
      1

[ ]:
```

Рис. 2.4: задание 1

2.7 2. Приведите свои примеры с выполнением операций над множествами элементов разных типов



```
Untitled.ipynb
[78]: x = ["apple", 40, 3.14]
[78]: 3-element Vector{Any}:
      "apple"
       40
      3.14
[79]: y = [40, "banana", 2.17]
[79]: 3-element Vector{Any}:
       40
      "banana"
       2.17
[80]: z = [3.14, "apple", "banana"]
[80]: 3-element Vector{Any}:
      3.14
      "apple"
      "banana"
[ ]:
```

Рис. 2.5: задание 2

```
[81]: # разность множеств

[82]: x_minus_y = setdiff(x,y)

[82]: 2-element Vector{Any}:
      "apple"
      3.14

[83]: x_minus_z = setdiff(x,z)

[83]: 1-element Vector{Any}:
      40

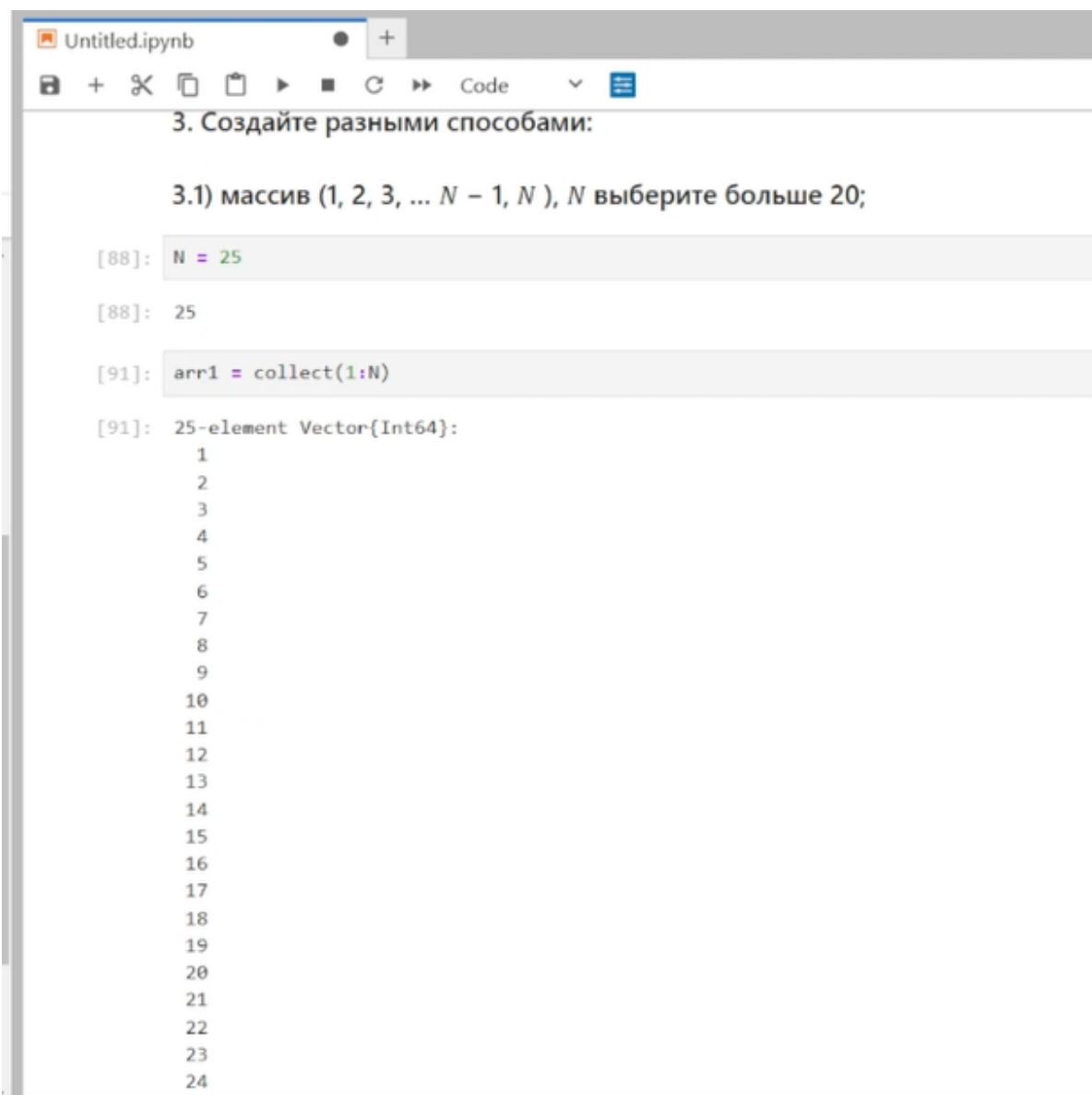
[84]: z_minus_y = setdiff(z,y)

[84]: 2-element Vector{Any}:
      3.14
      "apple"
```

Рис. 2.6: задание 2

2.8 3. Создайте разными способами:

2.9 3.1) массив $(1, 2, 3, \dots, N - 1, N)$, N выберите больше 20;

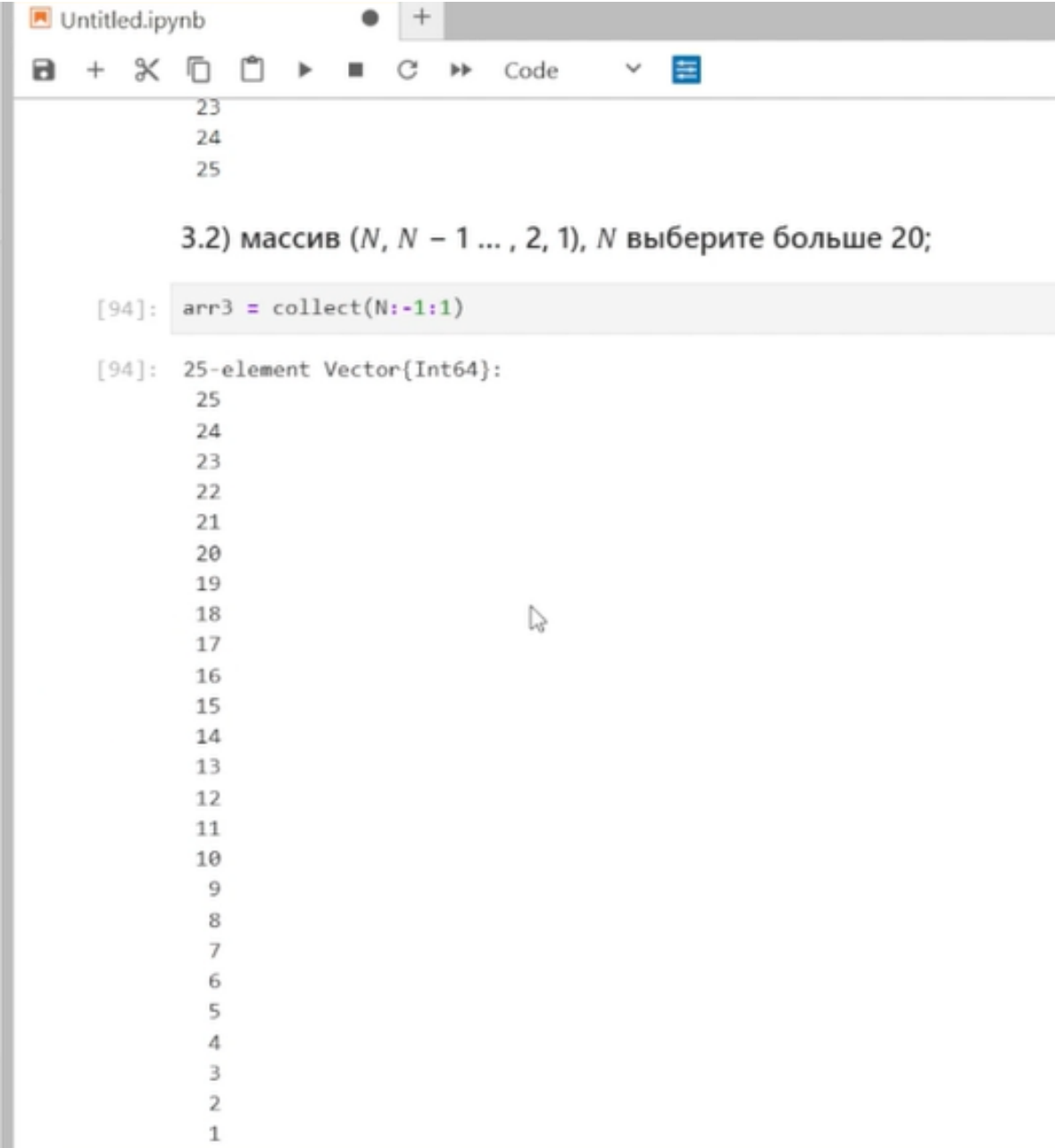


The screenshot shows a Jupyter Notebook window titled 'Untitled.ipynb'. The interface includes a toolbar with icons for saving, adding, deleting, and running code. The notebook content consists of two code cells. The first cell, labeled '[88]:', contains the code `N = 25` and the output is `25`. The second cell, labeled '[91]:', contains the code `arr1 = collect(1:N)` and the output is `25-element Vector{Int64}:` followed by a list of integers from 1 to 24.

```
[88]: N = 25
[88]: 25
[91]: arr1 = collect(1:N)
[91]: 25-element Vector{Int64}:
      1
      2
      3
      4
      5
      6
      7
      8
      9
     10
     11
     12
     13
     14
     15
     16
     17
     18
     19
     20
     21
     22
     23
     24
```

Рис. 2.7: задание 3.1

2.10 3.2) массив ($N, N - 1 \dots, 2, 1$), N выберите больше 20;



The screenshot shows a Jupyter Notebook interface with a file named 'Untitled.ipynb'. The code cell contains the command `arr3 = collect(N:-1:1)`. The output is a 25-element vector of integers, ranging from 25 down to 1. The notebook interface includes a toolbar with icons for saving, adding, deleting, and running code, as well as a 'Code' button.

```
23
24
25

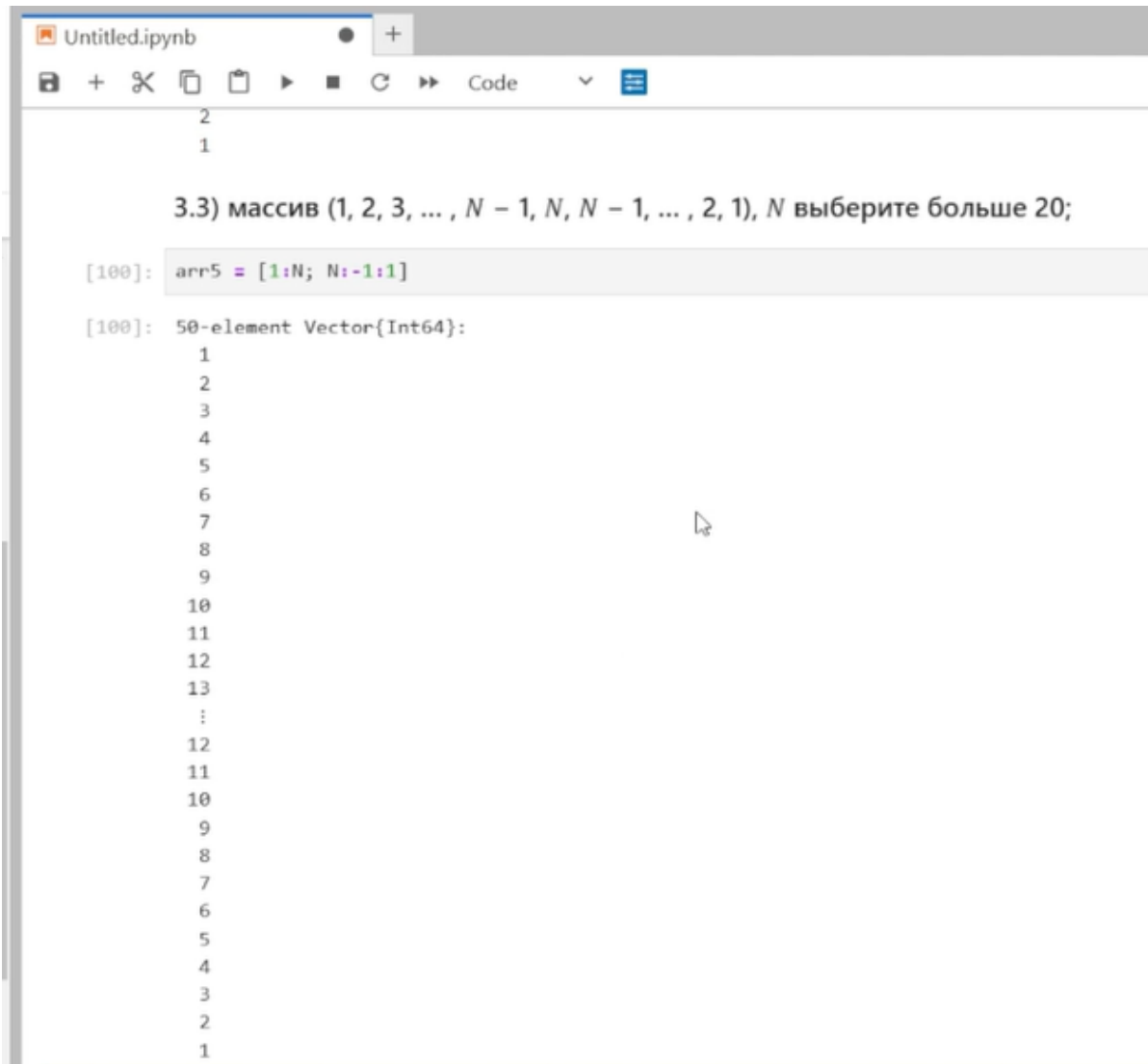
3.2) массив ( $N, N - 1 \dots, 2, 1$ ),  $N$  выберите больше 20;

[94]: arr3 = collect(N:-1:1)

[94]: 25-element Vector{Int64}:
      25
      24
      23
      22
      21
      20
      19
      18
      17
      16
      15
      14
      13
      12
      11
      10
       9
       8
       7
       6
       5
       4
       3
       2
       1
```

Рис. 2.8: задание 3.2

**2.11 3.3) массив (1, 2, 3, ..., N - 1, N, N - 1, ..., 2, 1), N
выберите больше 20;**



The screenshot shows a Jupyter Notebook interface with a single cell. The code in the cell is a C++ snippet that defines a vector named 'arr5' with 50 elements. The elements are integers from 1 to 20, followed by integers from 19 down to 1. The output of the code is displayed below the code cell, showing the 50 elements of the vector in a vertical list format.

```
2
1

3.3) массив (1, 2, 3, ..., N - 1, N, N - 1, ..., 2, 1), N выберите больше 20;

[100]: arr5 = [1:N; N:-1:1]

[100]: 50-element Vector{Int64}:
 1
 2
 3
 4
 5
 6
 7
 8
 9
10
11
12
13
⋮
12
11
10
 9
 8
 7
 6
 5
 4
 3
 2
 1
```

Рис. 2.9: задание 3.3

2.12 3.4) массив с именем tmp вида (4, 6, 3);

3.4) массив с именем tmp вида (4, 6, 3);

```
[101]: tmp = [4,6,3]
```

```
[101]: 3-element Vector{Int64}:  
      4  
      6  
      3
```

```
[103]: tmp = collect([4,6,3])
```

```
[103]: 3-element Vector{Int64}:  
      4  
      6  
      3
```

Рис. 2.10: задание 3.4

2.13 3.5) массив, в котором первый элемент массива tmp повторяется 10 раз;

3.5) массив, в котором первый элемент массива tmp повторяется 10 раз;

```
[05]: rep_tmp1 = fill(tmp[1],10)
```

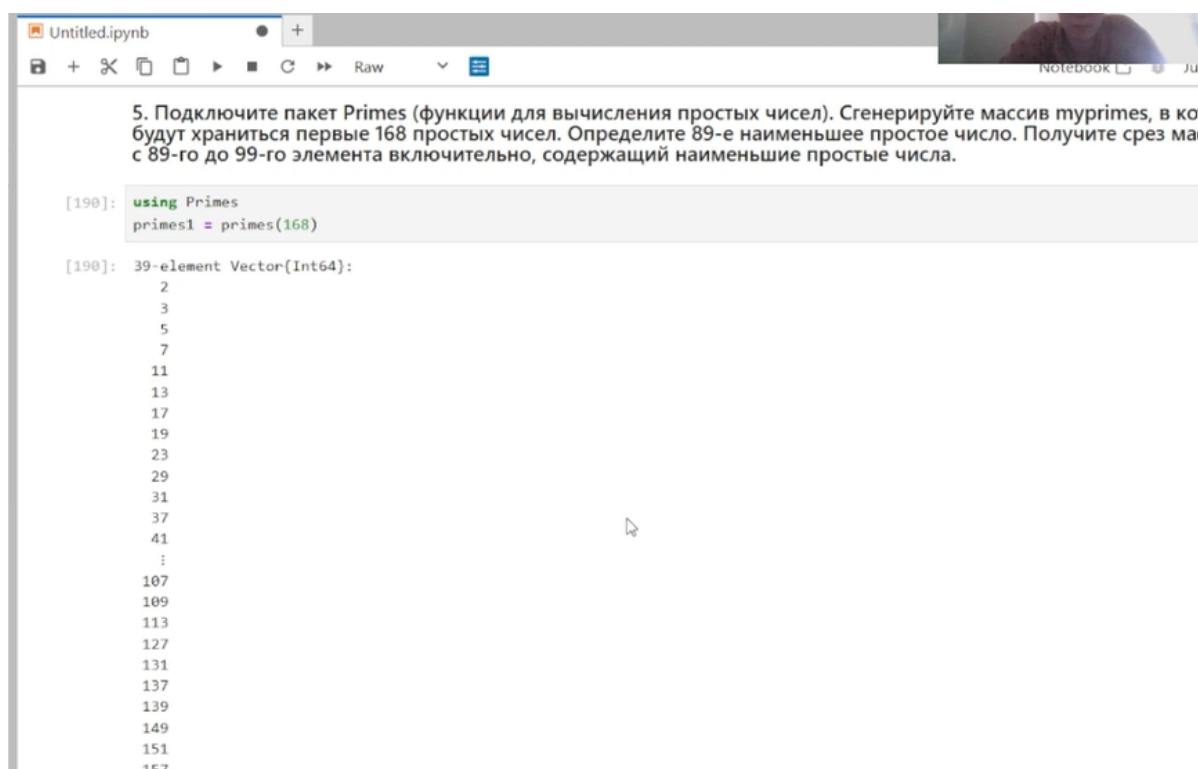
```
[05]: 10-element Vector{Int64}:
```

```
 4
 4
 4
 4
 4
 4
 4
 4
 4
 4
 4
```

Рис. 2.11: задание 3.5

(Заданий слишком много в пункте 3, поэтому все остальные задания находятся тут)

2.14 5. Подключите пакет Primes (функции для вычисления простых чисел). Сгенерируйте массив myprimes, в котором будут храниться первые 168 простых чисел. Определите 89-е наименьшее простое число. Получите срез массива с 89-го до 99-го элемента включительно, содержащий наименьшие простые числа.



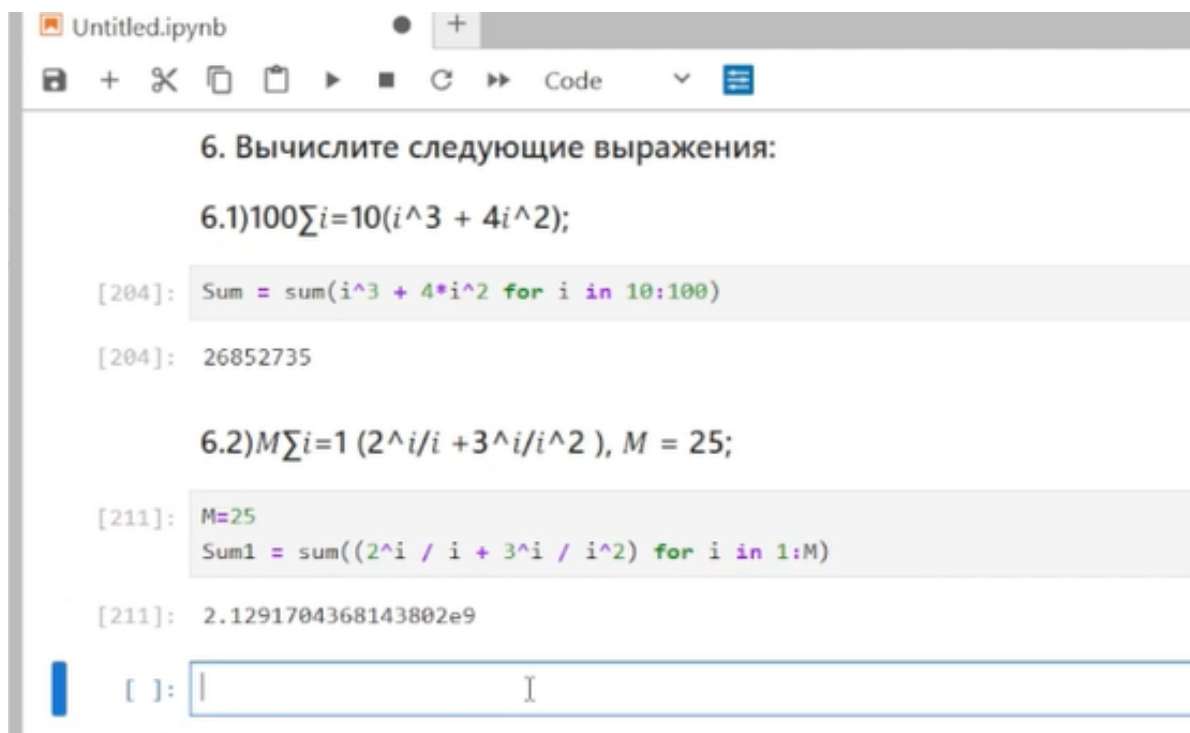
The screenshot shows a Jupyter Notebook window titled 'Untitled.ipynb'. The task description is written in Russian. Below it, the code for the task is shown, along with the output of the first code cell.

```
[190]: using Primes
       primes1 = primes(168)
```

```
[190]: 39-element Vector{Int64}:
        2
        3
        5
        7
       11
       13
       17
       19
       23
       29
       31
       37
       41
        ⋮
      107
      109
      113
      127
      131
      137
      139
      149
      151
      157
```

Рис. 2.12: задание 5

2.15 6. Вычислите следующие выражения



```
Untitled.ipynb
```

6. Вычислите следующие выражения:

6.1) $100 \sum_{i=10} (i^3 + 4i^2)$;

```
[204]: Sum = sum(i^3 + 4*i^2 for i in 10:100)
```

```
[204]: 26852735
```

6.2) $M \sum_{i=1} (2^i/i + 3^i/i^2)$, $M = 25$;

```
[211]: M=25
Sum1 = sum((2^i / i + 3^i / i^2) for i in 1:M)
```

```
[211]: 2.1291704368143802e9
```

```
[ ]: |
```

Рис. 2.13: задание 6

3 Выводы

Мы изучили несколько структур данных, реализованных в Julia, и научились применять их и операции над ними для решения задач