МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГБОУ ВО «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «МЭИ»

ИНЖЕНЕРНО-ЭКОНОМИЧЕСКИЙ ИНСТИТУТ

**ЛАБОРАТОРНАЯ РАБОТА**

по дисциплине «Объектно-ориентированный анализ и программирование»

на тему:

«Модули»

Работу выполнил:

Студент группы ИЭс-160п-19

Зубков Д. Ю.

Принял:

Преподаватель Овсянникова М. Р.

Москва

2021

**Оглавление**

[Условие задачи 3](#_Toc64849128)

[Метод решения задачи 3](#_Toc64849129)

[Алгоритм решения задачи 4](#_Toc64849130)

[Наборы тестовых данных 4](#_Toc64849131)

[**Файл «test\_inp\_1.txt»** 4](#_Toc64849132)

[**Файл «test\_inp\_2.txt»** 4](#_Toc64849133)

[**Файл «test\_inp\_typeerror.txt»** 4](#_Toc64849134)

[Состав данных 5](#_Toc64849135)

[Код программы 5](#_Toc64849136)

[Код модуля «unit.py» 5](#_Toc64849137)

[Код исполняемого файла «main.py» 6](#_Toc64849138)

[Тестирование и отладка 8](#_Toc64849139)

[Сценарий 1: Тестовые данные из файла 8](#_Toc64849140)

[**Тест 1:** Запуск с параметром командной строки «test\_inp\_1.txt» 8](#_Toc64849141)

[**Тест 2:** Запуск с параметром командной строки «test\_inp\_2.txt» 8](#_Toc64849142)

[**Тест 3:** Запуск с параметром командной строки «test\_inp\_typeerror.txt» 9](#_Toc64849143)

[Сценарий 2: ввод данных с клавиатуры 9](#_Toc64849144)

[**Тест 1:** Запуск без параметров командной строки 9](#_Toc64849145)

# Условие задачи

Вариант № 7

Использовать модуль проекта. В модуль проекта добавить функцию MIN, вычисляющую минимальный элемент вектора. Вычислить результат обработки векторов по формуле:

Где:

|  |  |
| --- | --- |
| Max | Максимальный элемент вектора |
| Min | Минимальный элемент вектора |
| S | Сумма положительных элементов вектора |
| K | Количество положительных элементов вектора |

# Метод решения задачи

Задача будет решаться методом декомпозиции задач на составные части, применением математических расчетов.

Формула для решения задачи:

Техническое выполнение задания и тестирование будет проводиться в следующих условиях:

|  |  |
| --- | --- |
| Язык программирования | Python 3.9 |
| Среда разработки | JetBrains PyCharm Community 2020.2.5 |
| Архитектура | Intel i386 (Core i9 9880H) |
| Операционная система | Apple macOS 11.2.1 |

# Алгоритм решения задачи



# Наборы тестовых данных

Тестовые данные представлены в виде трех .txt файлов.

### **Файл «test\_inp\_1.txt»**

|  |  |
| --- | --- |
| **Тестовые данные** | **Ожидаемый результат** |
| 1.01 -4.2 0.83 -9.64 12  14 9.25 -6 8.28  3.9 -8.21 7.4 0.26 5.45 -9.8 | -0.93 |

### **Файл «test\_inp\_2.txt»**

|  |  |
| --- | --- |
| **Тестовые данные** | **Ожидаемый результат** |
| 11.4 0.02 -3  4.14 -2.45 3.16 -4.25 0  2.99 -3.18 2.17 -5 -1.04 0.5 | 2.60 |

### **Файл «test\_inp\_typeerror.txt»**

|  |  |
| --- | --- |
| **Тестовые данные** | **Ожидаемый результат** |
| 1 2 3  4 k 6  9 8 7 | Сообщение об ошибке |

# Состав данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Класс** | **Имя** | **Тип** | **Структура** | **Смысл** |
| **Входные данные** | vectors | Вещественный | Двумерный массив | Массив исходных векторов |
| **Выходные данные** | result | Вещественный | Простая переменная | Результат вычисления по формуле |
| **Промежуточные данные** | sa | Вещественный | Простая переменная | Сумма положительных элементов 1‑го вектора |
| kb | Вещественный | Простая переменная | Количество положительных элементов 1‑го вектора |
|  | min\_c | Вещественный | Простая переменная | Минимальный элемент 3‑го вектора |
|  | sc | Вещественный | Простая переменная | Сумма положительных элементов 1‑го вектора |
|  | kc | Вещественный | Простая переменная | Количество положительных элементов 2‑го вектора |

# Код программы

## Код модуля «unit.py»

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54 | **def** check\_floats(arr):  return all(isinstance(x, float) for x in arr)  **def** get\_max(arr):  if not check\_floats(arr):  return None  try:  m = arr[0]  except IndexError:  return None  for n in arr:  if n > m:  m = n  return m  **def** get\_min(arr):  if not check\_floats(arr):  return None  try:  m = arr[0]  except IndexError:  return None  for n in arr:  if n < m:  m = n  return m  **def** get\_s(arr):  if not check\_floats(arr):  return None  try:  s = 0  except IndexError:  return None  for n in arr:  if n > 0:  s += n  return s  **def** get\_k(arr):  if not check\_floats(arr):  return None  try:  k = 0  except IndexError:  return None  for n in arr:  if n > 0:  k += 1  return k |

## Код исполняемого файла «main.py»

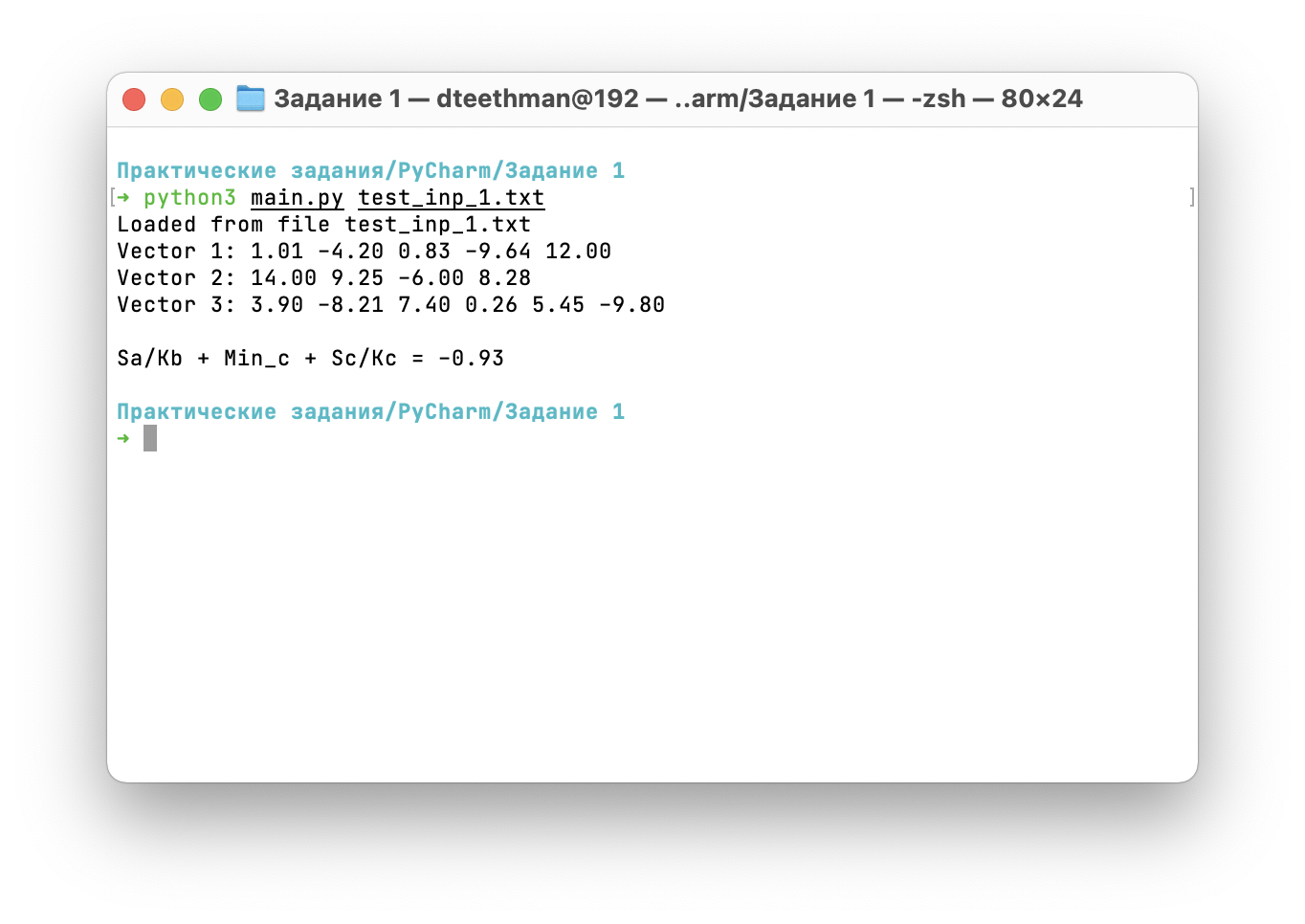
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84 | import unit  from sys import argv  **def** make\_floats\_array(str\_arr):  try:  a = [float(num) for num in str\_arr]  except ValueError:  print('Error: can’t make array of floats cause of wrong input')  exit(0)  return a  **def** input\_from\_keyboard():  print('Input file not specified or not exist\n'  'Type 3 arrays from keyboard in single line divided by return:')  v = []  for i in range (3):  str\_arr = input().strip().split(' ')  arr = make\_floats\_array(str\_arr)  v.append(arr)  if v:  return v  else:  return None  **def** load\_from\_file(filename):  try:  f = open(filename, 'r')  except FileNotFoundError:  return input\_from\_keyboard()  v\_raw = f.read().split('\n')  f.close()  v = []  for i in range(len(v\_raw)):  v.append(make\_floats\_array(v\_raw[i].strip().split(' ')))  if v:  print(**f**'Loaded from file {filename}')  print(show\_vectors(v))  return v  else:  return None  **def** show\_vectors(arr):  text = ''  for i in range(len(arr)):  vector\_str = ' '.join([ **f**'{num**:.2f**}' for num in arr[i]])  text += **f**'Vector {i + 1}: {vector\_str}\n'  return text  **def** save\_to\_file(filename, text):  f = open(filename, 'w+', newline='\n')  f.write(text)  f.close()  if \_\_name\_\_ == '\_\_main\_\_':  vectors = []  if len(argv) < 2:  vectors = input\_from\_keyboard()  else:  vectors = load\_from\_file(argv[1])  if vectors and len(vectors) == 3:  sa = unit.get\_s(vectors[0])  kb = unit.get\_k(vectors[1])  min\_c = unit.get\_min(vectors[2])  sc = unit.get\_s(vectors[2])  kc = unit.get\_k(vectors[2])  try:  result = sa / kb + min\_c + sc / kc  except ZeroDivisionError:  print('Error: divided by zero')  exit(0)  except TypeError:  print('Error: bad input value')  exit(0)  res\_str = **f**'Sa/Kb + Min\_c + Sc/Kc = {result**:.2f**}'  print(res\_str)  save\_to\_file('out.txt', **f**'{show\_vectors(vectors)}\n{res\_str}')  else:  print('Error: loaded more or less that 3 vectors') |

# Тестирование и отладка

## Сценарий 1: Тестовые данные из файла

### **Тест 1:** Запуск с параметром командной строки «test\_inp\_1.txt»

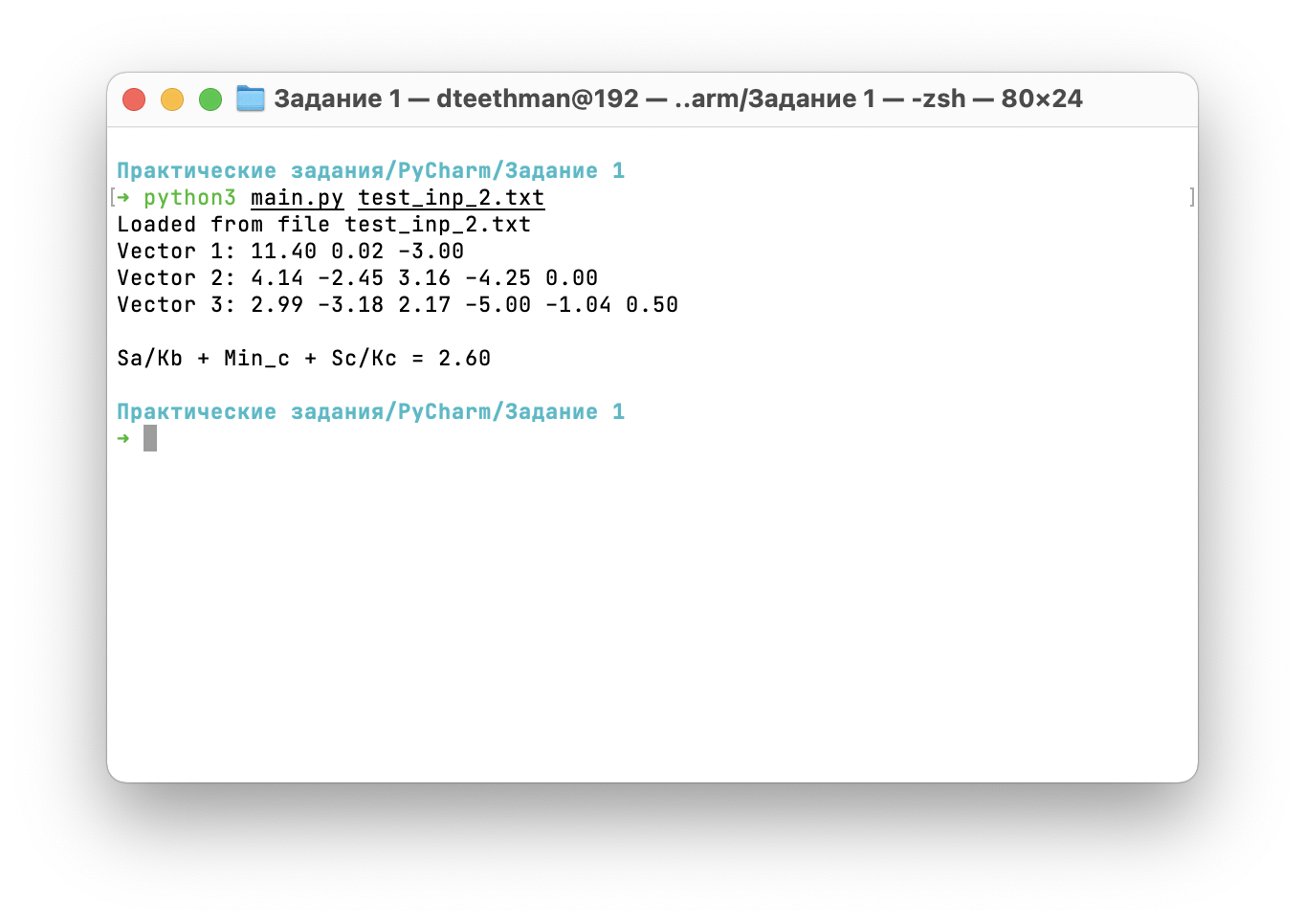
**Ожидаемый результат:** -0.934166666666667

****

**Вывод:** тест пройден.

### **Тест 2:** Запуск с параметром командной строки «test\_inp\_2.txt»

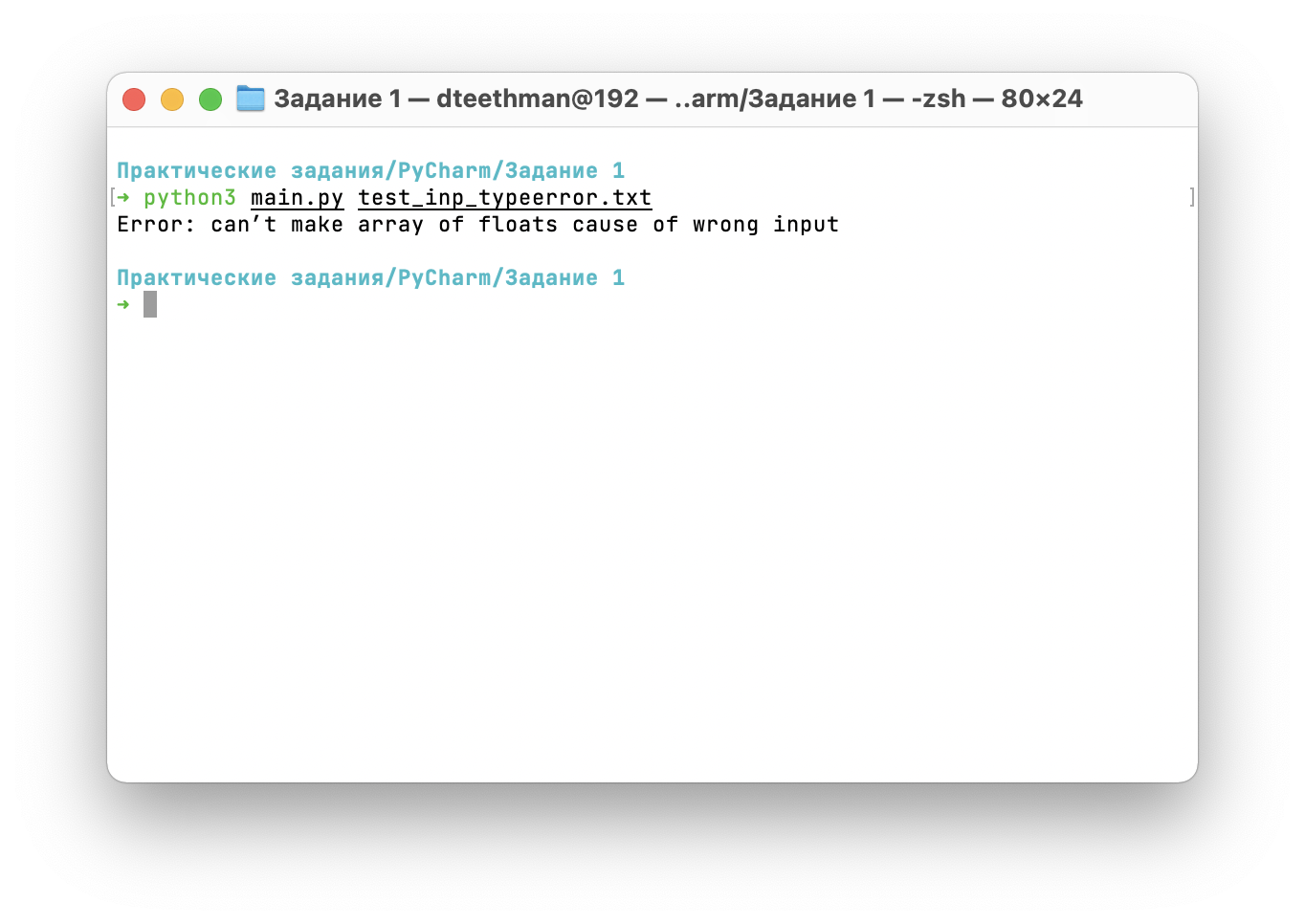
**Ожидаемый результат:** -2.5966666666666667

****

**Вывод:** тест пройден.

### **Тест 3:** Запуск с параметром командной строки «test\_inp\_typeerror.txt»

**Ожидаемый результат:** сообщение об ошибке

****

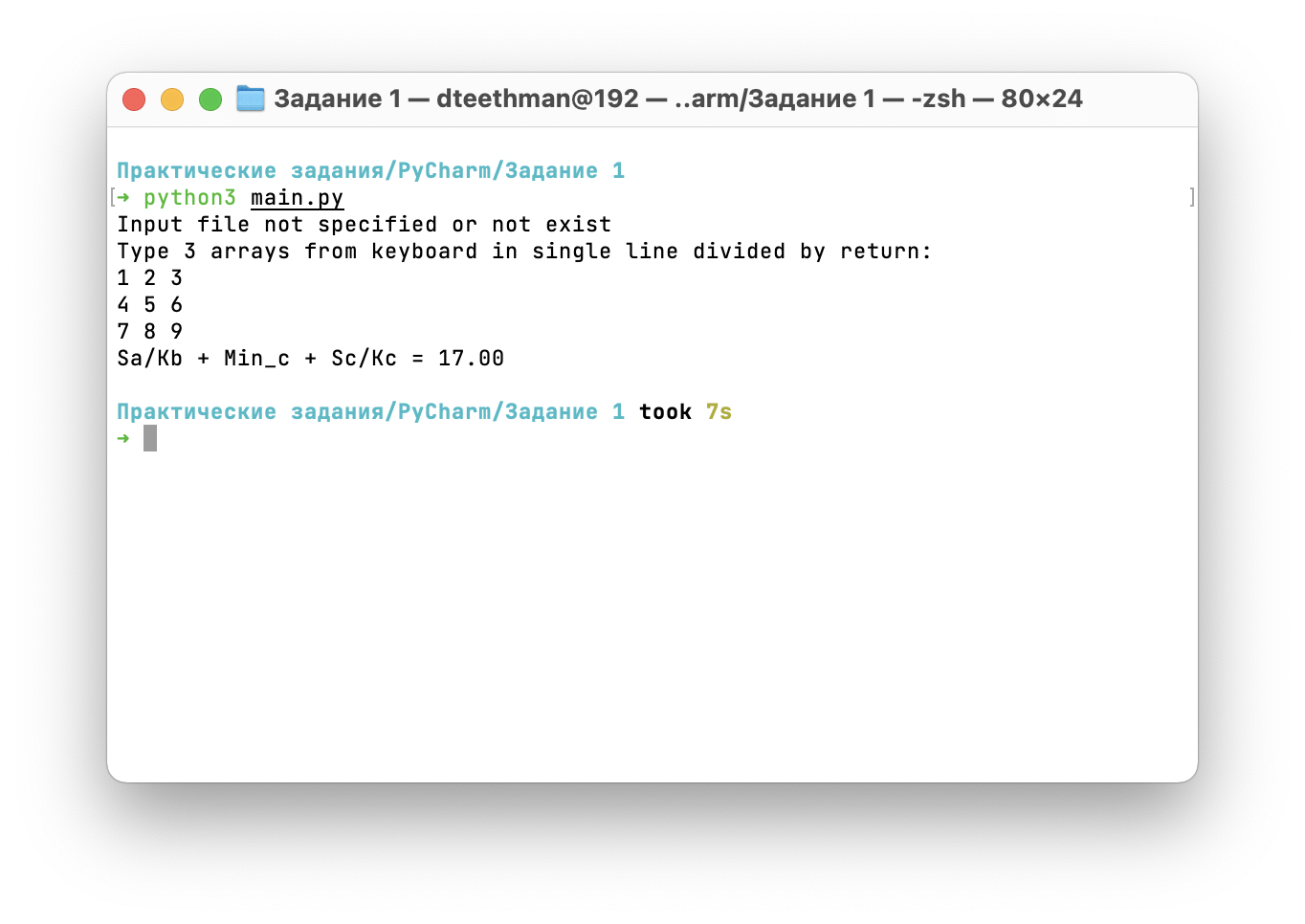
**Вывод:** тест пройден.

## Сценарий 2: ввод данных с клавиатуры

### **Тест 1:** Запуск без параметров командной строки

Вектора: [1, 2, 3], [4, 5, 6], [7, 8, 9]

**Ожидаемый результат:** 17

****

**Вывод:** тест пройден.