

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

«Работа с функциями в языке Python»

ОТЧЕТ
по лабораторной работе №11
дисциплины
«Основы программной инженерии»

Выполнил:

Сотников Андрей Александрович
2 курс, группа ПИЖ-б-о-21-1,
09.03.04 «Программная
инженерия», направленность
(профиль) «Разработка и
сопровождение программного
обеспечения», очная форма
обучения

(подпись)

Проверил:

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2022 г.

Проработка примера из лабораторной работы:

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  import sys
4  from datetime import date
5
6
7  AndrejMirrox *
8  def get_worker():
9      """
10     Запросить данные о работнике.
11     """
12     name = input("Фамилия и инициалы? ")
13     post = input("Должность? ")
14     year = int(input("Год поступления? "))
15
16     # Создать словарь.
17     return {
18         'name': name,
19         'post': post,
20         'year': year,
21     }
22
23  AndrejMirrox *
24  def display_workers(staff):
25      """
26      Отобразить список работников.
27      """
28      # Проверить, что список работников не пуст.
29      if staff:
30          # Заголовок таблицы.
31          line = '+--{}+--{}+--{}+--{}+'.format(
32              '-' * 4,
33              '-' * 30,
34              '-' * 20,
35              '-' * 8
36          )
37          print(line)
38          print(
39              '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
40                  "No",
41                  "Ф.И.О.",
42                  "Должность",
43                  "Год"
44              )
45          )
46          print(line)
```

Рисунок 1 – Код примера

```

47
48     # Вывести данные о всех сотрудниках.
49     for idx, worker in enumerate(staff, 1):
50         print(
51             '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
52                 idx,
53                 worker.get('name', ''),
54                 worker.get('post', ''),
55                 worker.get('year', 0)
56             )
57         )
58     print(line)
59 else:
60     print("Список работников пуст.")
61
62
63     AndrejMirrox *
64 def select_workers(staff, period):
65     """
66     Выбрать работников с заданным стажем.
67     """
68     # Получить текущую дату.
69     today = date.today()
70
71     # Сформировать список работников.
72     result = []
73     for employee in staff:
74         if today.year - employee.get('year', today.year) >= period:
75             result.append(employee)
76
77     # Возвратить список выбранных работников.
78     return result
79
80     AndrejMirrox *
81 def main():
82     """
83     Главная функция программы.
84     """
85     # Список работников.
86     workers = []
87
88     # Организовать бесконечный цикл запроса команд.
89     while True:
90
91         # Запросить команду из терминала.
92         command = input(">>> ").lower()
93

```

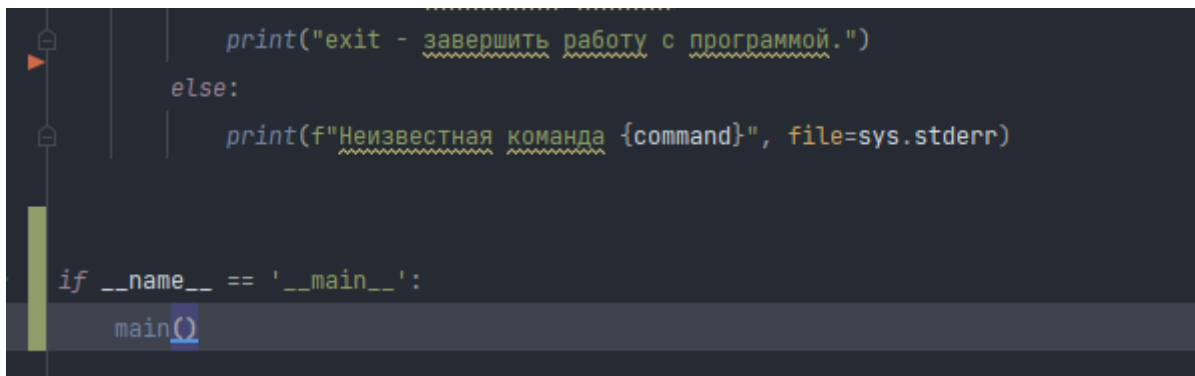
Рисунок 2 – Код примера

```

87
88     # Организовать бесконечный цикл запроса команд.
89     while True:
90
91         # Запросить команду из терминала.
92         command = input(">>> ").lower()
93
94         # Выполнить действие в соответствие с командой.
95         if command == 'exit':
96             break
97
98         elif command == 'add':
99
100             # Запросить данные о работнике.
101             worker = get_worker()
102
103             # Добавить словарь в список.
104             workers.append(worker)
105
106             # Отсортировать список в случае необходимости.
107             if len(workers) > 1:
108                 workers.sort(key=lambda item: item.get('name', ''))
109
110             elif command == 'list':
111                 # Отобразить всех работников.
112                 display_workers(workers)
113
114             elif command.startswith('select '):
115
116                 # Разбить команду на части для выделения стажа.
117                 parts = command.split(' ', maxsplit=1)
118
119                 # Получить требуемый стаж.
120                 period = int(parts[1])
121
122                 # Выбрать работников с заданным стажем.
123                 selected = select_workers(workers, period)
124
125                 # Отобразить выбранных работников.
126                 display_workers(selected)
127
128             elif command == 'help':
129
130                 # Вывести справку о работе с программой.
131                 print("Список команд:\n")
132                 print("add - добавить работника;")
133                 print("list - вывести список работников;")
134                 print("select <стаж> - запросить работников со стажем;")
135                 print("help - отобразить справку;")
136                 print("exit - завершить работу с программой.")
137
138             else:

```

Рисунок 3 – Код примера



```
print("exit - завершить работу с программой.")
else:
    print(f"Неизвестная команда {command}", file=sys.stderr)

if __name__ == '__main__':
    main()
```

Рисунок 4 – Код примера

```

C:\Users\sotni\AppData\Local\Programs\Python\Python39\python.exe C:\labor-11\
>>> help
Список команд:

add - добавить работника;
list - вывести список работников;
select <стаж> - запросить работников со стажем;
help - отобразить справку;
exit - завершить работу с программой.
>>> add
Фамилия и инициалы? Сотников
Должность? студент
Год поступления? 2021
>>> add
Фамилия и инициалы? Человек
Должность? уборщик
Год поступления? 2030
>>> list
+-----+-----+-----+-----+
| No |          Ф.И.О.          |      Должность      |   Год   |
+-----+-----+-----+-----+
|   1 | Сотников                | студент              | 2021    |
|   2 | Человек                  | уборщик              | 2030    |
+-----+-----+-----+-----+
>>> select 1
+-----+-----+-----+-----+
| No |          Ф.И.О.          |      Должность      |   Год   |
+-----+-----+-----+-----+
|   1 | Сотников                | студент              | 2021    |
+-----+-----+-----+-----+
>>>

```

Рисунок 5 – Результат работы примера

Задание №1: решить следующую задачу: основная ветка программы, не считая заголовков функций, состоит из двух строки кода. Это вызов функции `test()` и инструкции `if __name__ == '__main__':`. В ней запрашивается на ввод целое число. Если оно положительное, то вызывается функция `positive()`, тело которой содержит команду вывода на экран слова "Положительное". Если число отрицательное, то вызывается функция `negative()`, ее тело содержит выражение вывода на экран слова "Отрицательное". Понятно, что вызов `test()` должен следовать после определения функций. Однако имеет ли значение порядок определения самих функций? То есть должны ли определения `positive()` и `negative()` предшествовать `test()` или могут следовать после него? Проверьте вашу гипотезу, поменяв объявления функций местами. Попробуйте объяснить результат.



```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  new *
5  def test():
6      num = int(input("Введите число: "))
7      if num > 0:
8          positive()
9      elif num < 0:
10         negative()
11     else:
12         print("Число равно нулю")
13
14  new *
15  def positive():
16     print("Число положительное")
17
18  new *
19  def negative():
20     print("Число отрицательное")
21
22  if __name__ == "__main__":
23     test()
24
negative()
ex x 1_task x
C:\Users\sotni\AppData\Local\Programs\Python\Python39\python.exe C:\labor-11\PyCharm\1_task.py
Введите число: 13
Число положительное
```

Рисунок 6 – Код и результат работы программы задания №1 с вызовом функций `positive()` и `negative()` после `test()`

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  new *
5  def positive():
6      print("Число положительное")
7
8  new *
9  def negative():
10     print("Число отрицательное")
11
12     new *
13     def test():
14         num = int(input("Введите число: "))
15         if num > 0:
16             positive()
17         elif num < 0:
18             negative()
19         else:
20             print("Число равно нулю")
21
22     if __name__ == "__main__":
23         test()
```

ex x 1_task x

C:\Users\sotni\AppData\Local\Programs\Python\Python39\python.exe C:\labor-11\PyC

Введите число: -15

Число отрицательное

Рисунок 7 – Код и результат работы программы задания №1 с вызовом функций positive() и negative() перед test()

Задание №2: Решите следующую задачу: в основной ветке программы вызывается функция `cylinder()`, которая вычисляет площадь цилиндра. В теле `cylinder()` определена функция `circle()`, вычисляющая площадь круга по формуле $S = \pi r^2$. В теле `cylinder()` у пользователя спрашивается, хочет ли он получить только площадь боковой поверхности цилиндра, которая вычисляется по формуле $S_{\text{бок}} = 2\pi r h$, или полную площадь цилиндра. В последнем случае к площади боковой поверхности цилиндра должен добавляться удвоенный результат вычислений функции `circle()`.

```
6
7 def cylinder():
8     r = float(input("Введите радиус: "))
9     h = float(input("Введите высоту: "))
10    s_side = 2 * math.pi * r * h
11
12    new *
13    def circle():
14        s_circle = math.pi * r ** 2
15        return s_circle
16
17    check = input("Введите Y для бок. площ. или N для всей площ.: ")
18    if check == "Y":
19        print(f"Бок. площ. цилиндра: {s_side}")
20    else:
21        full_area = s_side + circle() * 2
22        print(f"Полная площ. цилиндра: {full_area}")
23
24 if __name__ == "__main__":
25     cylinder()
```

cylinder()

2_task x

C:\Users\sotni\AppData\Local\Programs\Python\Python39\python.exe C:\labor

Введите радиус: 20

Введите высоту: 42

Введите Y для бок. площ. или N для всей площ.: Y

Бок. площ. цилиндра: 5277.875658030853

Рисунок 8 – Код и результат работы программы задания №2

```
1  ▶  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import math
5
6
7  def cylinder():
8      r = float(input("Введите радиус: "))
9      h = float(input("Введите высоту: "))
10     s_side = 2 * math.pi * r * h
11
12     new *
13     def circle():
14         s_circle = math.pi * r ** 2
15         return s_circle
16
17     check = input("Введите Y для бок. площ. или N для всей площ.: ")
18     if check == "Y":
19         print(f"Бок. площ. цилиндра: {s_side}")
20     else:
21         full_area = s_side + circle() * 2
22         print(f"Полная площ. цилиндра: {full_area}")
23
24     if __name__ == "__main__":
25         cylinder()
```

cylinder()

2_task ×

C:\Users\sotni\AppData\Local\Programs\Python\Python39\python.exe C:\labor-1

Введите радиус: 20

Введите высоту: 42

Введите Y для бок. площ. или N для всей площ.: N

Полная площ. цилиндра: 7791.149780902688

Рисунок 9 – Код и результат работы программы задания №2

Задание №3: решите следующую задачу: напишите функцию, которая считывает с клавиатуры числа и перемножает их до тех пор, пока не будет введен 0. Функция должна возвращать полученное произведение. Вызовите функцию и выведите на экран результат ее работы.

```
1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4      def multiply():
5          f_num = 1
6          while True:
7              n_num = int(input("Введите числа: "))
8              if n_num != 0:
9                  f_num *= n_num
10             else:
11                 break
12         return f_num
13
14
15  ▶  if __name__ == "__main__":
16      print(f"Числа умноженные до '0': {multiply()}")
```

if __name__ == "__main__"

3_task ×

C:\Users\sotni\AppData\Local\Programs\Python\Python39\pytho

Введите числа: 6

Введите числа: 8

Введите числа: 6

Введите числа: 1

Введите числа: 2

Введите числа: 15

Введите числа: 44

Введите числа: 2

Введите числа: 0

Числа умноженные до '0': 760320

Рисунок 10 – Код и результат работы программы задания №3

Задание №4: решите следующую задачу: напишите программу, в которой определены следующие четыре функции:

1. Функция `get_input()` не имеет параметров, запрашивает ввод с клавиатуры и возвращает в основную программу полученную строку.

2. Функция `test_input()` имеет один параметр. В теле она проверяет, можно ли переданное ей значение преобразовать к целому числу. Если можно, возвращает логическое `True`. Если нельзя – `False`.

3. Функция `str_to_int()` имеет один параметр. В теле преобразовывает переданное значение к целочисленному типу. Возвращает полученное число.

4. Функция `print_int()` имеет один параметр. Она выводит переданное значение на экран и ничего не возвращает.

В основной ветке программы вызовите первую функцию. То, что она вернула, передайте во вторую функцию. Если вторая функция вернула `True`, то те же данные (из первой функции) передайте в третью функцию, а возвращенное третьей функцией значение – в четвертую.

```
1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4      def get_input():
5          return input("Введите число: ")
6
7
8      def test_input(num):
9          try:
10             int(num)
11             return True
12         except ValueError:
13             return False
14
15
16      def str_to_int(num):
17          return int(num)
18
19
20      def print_int(num):
21          print(num, type(num))
22
23
24  ▶  if __name__ == "__main__":
25          number = get_input()
26          print(number, type(number))
27          if test_input(number):
28              str_to_int(number)
29              print_int(str_to_int(number))
30          else:
31              print(f"Нельзя преобразовать в число {number}")
```

print_int()

4_task ×

C:\Users\sotni\AppData\Local\Programs\Python\Python39\python.exe C:\labor

Введите число: лол кек чебурек

лол кек чебурек <class 'str'>

Нельзя преобразовать в число лол кек чебурек

Рисунок 11 – Код и результат работы программы задания №4

```
1  ▶  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  def get_input():
5      return input("Введите число: ")
6
7
8  def test_input(num):
9      try:
10         int(num)
11         return True
12     except ValueError:
13         return False
14
15
16  def str_to_int(num):
17      return int(num)
18
19
20  def print_int(num):
21      print(num, type(num))
22
23
24  ▶  if __name__ == "__main__":
25      number = get_input()
26      print(number, type(number))
27      if test_input(number):
28          str_to_int(number)
29          print_int(str_to_int(number))
30      else:
31          print(f"Нельзя преобразовать в число {number}")
```

print_int()

4_task ×

C:\Users\sotni\AppData\Local\Programs\Python\Python39\python.exe C:\labor

Введите число: 452

452 <class 'str'>

452 <class 'int'>

Рисунок 12 – Код и результат работы программы задания №4

Индивидуальное задание: решить индивидуальное задание лабораторной работы 2.6, оформив каждую команду в виде отдельной функции.

```

1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4      import sys
5      from datetime import date
6
7
8      new *
9      def add_man():
10         """
11         Добавление людей
12         """
13
14         name = input("Фамилия и инициалы? ")
15         post = input("Телефон? ")
16         year = input("Год рождения? ")
17         year = year.split(".")
18         year = date(int(year[0]), int(year[1]), int(year[2]))
19
20         # Создать словарь.
21         man = {
22             'name': name,
23             'tel': post,
24             'date': year,
25         }
26
27         # Добавить словарь в список.
28         return man
29
30
31     new *
32     def list_man(people):
33         """
34         Вывод людей
35         """
36
37         line = '+--{}-+-{}-+-{}-+-{}-+'.format(
38             '-' * 4,
39             '-' * 30,
40             '-' * 20,
41             '-' * 12
42         )
43         print(line)
44         print(
45             '| {:^4} | {:^30} | {:^20} | {:^12} |'.format(

```

Рисунок 13 – Код программы для индивидуального задания


```

41     )
42     print(line)
43     print(
44         '| {:^4} | {:^30} | {:^20} | {:^12} |'.format(
45             "№",
46             "Ф.И.О.",
47             "Телефон",
48             "Год рождения"
49         )
50     )
51     print(line)
52
53     for idx, man in enumerate(people, 1):
54         print(
55             '| {:>4} | {:<30} | {:<20} | {:>12} |'.format(
56                 idx,
57                 man.get('name', ''),
58                 man.get('tel', ''),
59                 str(man.get('date', 0))
60             )
61         )
62     print(line)
63
64     new *
65     def select_man(person):
66         """
67         Вывод конкретных людей
68         """
69
70         count = 0
71         for man in people:
72             if man.get('name', person).lower() == person.lower():
73                 count += 1
74                 line = '+-{}-+-{}-+-{}-+-{}-+'.format(
75                     '-' * 4,
76                     '-' * 30,
77                     '-' * 20,
78                     '-' * 12
79                 )
80                 print(line)
81                 print(
82                     '| {:^4} | {:^30} | {:^20} | {:^12} |'.format(
83                         "№",
84                         "Ф.И.О.",
85                         "Телефон",
86                         "Год рождения"
87                     )
88                 )

```

Рисунок 14 – Код программы для индивидуального задания

```

        print(line)
        print(
            '| {:>4} | {:<30} | {:<20} | {:>12} |'.format(
                count,
                man.get('name', ''),
                man.get('tel', ''),
                str(man.get('date', 0))
            )
        )
        print(line)

    if count == 0:
        print("Люди с заданным именем не найдены.")

new *
def help_man():
    print("Список команд:\n")
    print("add - добавить человека;")
    print("list - вывести список людей;")
    print("select <имя> - запросить людей с этим именем;")
    print("help - отобразить справку;")
    print("exit - завершить работу с программой.")

if __name__ == '__main__':
    # Список работников.
    people = []

    while True:
        command = input(">>> ").lower()

        if command == 'exit':
            break

        elif command == "add":
            people.append(add_man())
            if len(people) > 1:
                people.sort(key=lambda item: item.get('tel', ''))

```

Рисунок 15 – Код программы для индивидуального задания

```
29     elif command == 'list':
30         | list_man(people)
31
32     elif command.startswith('select'):
33         | parts = command.split(' ', maxsplit=1)
34         | period = parts[1]
35         | select_man(period)
36
37     elif command == 'help':
38         | help_man()
39
40     else:
41         | print(f"Неизвестная команда {command}", file=sys.stderr)
```

Рисунок 16 – Код программы для индивидуального задания

```

C:\Users\sotni\AppData\Local\Programs\Python\Python39\python.exe C:\labor-11\PyCharm\In
>>> add
Фамилия и инициалы? Сотников
Телефон? 89283486939
Год рождения? 2003.10.02
>>> add
Фамилия и инициалы? Человек
Телефон? 89996662211
Год рождения? 1950.11.15
>>> list
+-----+-----+-----+
| № |          Ф.И.О.          |      Телефон      | Год рождения |
+-----+-----+-----+
|  1 | Сотников                | 89283486939       | 2003-10-02 |
|  2 | Человек                  | 89996662211       | 1950-11-15 |
+-----+-----+-----+
>>> select Человек
+-----+-----+-----+
| № |          Ф.И.О.          |      Телефон      | Год рождения |
+-----+-----+-----+
|  1 | Человек                  | 89996662211       | 1950-11-15 |
+-----+-----+-----+
>>> help
Список команд:

add - добавить человека;
list - вывести список людей;
select <имя> - запросить людей с этим именем;
help - отобразить справку;
exit - завершить работу с программой.

```

Рисунок 17 – Результат работы программы для индивидуального задания

Контрольные вопросы

1. Каково назначение функций в языке программирования Python?

Функция в программировании представляет собой обособленный участок кода, который можно вызывать, обратившись к нему по имени, которым он был назван. При вызове происходит выполнение команд тела функции.

2. Каково назначение операторов def и return ?

В языке программирования Python функции определяются с помощью оператора def

Функции могут передавать какие-либо данные из своих тел в основную ветку программы. Говорят, что функция возвращает значение. В большинстве языков программирования, в том числе Python, выход из функции и передача данных в то место, откуда она была вызвана, выполняется оператором return.

3. Каково назначение локальных и глобальных переменных при написании функций в Python?

В программировании особое внимание уделяется концепции о локальных и глобальных переменных, а также связанное с ними представление об областях видимости. Соответственно, локальные переменные видны только в локальной области видимости, которой может выступать отдельно взятая функция. Глобальные переменные видны во всей программе. "Видны" – значит, известны, доступны. К ним можно обратиться по имени и получить связанное с ними значение.

4. Как вернуть несколько значений из функции Python?

```
return side, full
```

```
scyl, fcyl = cylinder()
```

5. Какие существуют способы передачи значений в функцию?

Через параметры, и через ввод, запрашиваемый самой функцией

6. Как задать значение аргументов функции по умолчанию?

```
def cylinder(h, r=1):
```

7. Каково назначение lambda-выражений в языке Python?

Python поддерживает интересный синтаксис, позволяющий определять небольшие однострочные функции на лету. Позаимствованные из Lisp, так называемые lambda-функции могут быть использованы везде, где требуется функция

8. Как осуществляется документирование кода согласно PEP257?

Документирование кода в python - достаточно важный аспект, ведь от нее порой зависит читаемость и быстрота понимания вашего кода, как другими людьми, так и вами через полгода. PEP 257 описывает соглашения, связанные со строками документации python, рассказывает о том, как нужно документировать python код. Цель этого PEP - стандартизировать структуру строк документации: что они должны в себя включать, и как это написать (не касаясь вопроса синтаксиса строк документации). Этот PEP описывает соглашения, а не правила или синтаксис.

9. В чем особенность однострочных и многострочных форм строк документации?

Для согласованности, всегда используйте `"""triple double quotes"""` для строк документации. Используйте `r"""raw triple double quotes"""`, если вы будете использовать обратную косую черту в строке документации. Существует две формы строк документации: однострочная и многострочная.