

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»  
ИНСТИТУТ ЦИФРОВОГО РАЗВИТИЯ

Отчет о лабораторной работе №11 по дисциплине «Основы  
программной инженерии»

Выполнил:  
Мамонтов Д.В.,  
2 курс, группа ПИЖ-б-о-20-1,  
Проверил:  
Доцент кафедры инфокоммуникаций,  
Воронкин Р.А.

Ставрополь, 2022 г

## ХОД РАБОТЫ

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3
4  import sys
5  from datetime import date
6
7
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     # Создать словарь.
16     return {
17         'name': name,
18         'post': post,
19         'year': year,
20     }
21
22
23 def display_workers(staff):
24     """
25     Отобразить список работников.
26     """
27     # Проверить, что список работников не пуст.
28     if staff:
29         # Заголовок таблицы.
30         line = '+-{}-+-{}-+-{}-+-{}-+'.format(
31             '-' * 4,
```

Рисунок 1 – код программы

```
31             '-' * 4,
32             '-' * 30,
33             '-' * 20,
34             '-' * 8
35         )
36         print(line)
37         print(
38             '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
39                 "No",
40                 "Ф.И.О.",
41                 "Должность",
42                 "Год"
43             )
44         )
45         print(line)
46         # Вывести данные о всех сотрудниках.
47         for idx, worker in enumerate(staff, 1):
48             print(
49                 '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
50                     idx,
51                     worker.get('name', ''),
52                     worker.get('post', ''),
53                     worker.get('year', 0)
54                 )
55             )
56         print(line)
57     else:
58         print("Список работников пуст.")
59
60
61 def select_workers(staff, period):
```

Рисунок 2 – код программы (продолжение)

```

62     """
63     Выбрать работников с заданным стажем.
64     """
65     # Получить текущую дату.
66     today = date.today()
67     # Сформировать список работников.
68     result = []
69     for employee in staff:
70         if today.year - employee.get('year', today.year) >= period:
71             result.append(employee)
72     # Возвратить список выбранных работников.
73     return result
74
75
76 def main():
77     """
78     Главная функция программы.
79     """
80     # Список работников.
81     workers = []
82     # Организовать бесконечный цикл запроса команд.
83     while True:
84         # Запросить команду из терминала.
85         command = input(">>> ").lower()
86         # Выполнить действие в соответствии с командой.
87         if command == 'exit':
88             break
89         elif command == 'add':
90             # Запросить данные о работнике.
91             worker = get_worker()
92             # Добавить словарь в список.

```

Рисунок 3 – код программы (продолжение)

```

93         workers.append(worker)
94         # Отсортировать список в случае необходимости.
95         if len(workers) > 1:
96             workers.sort(key=lambda item: item.get('name', ''))
97         elif command == 'list':
98             # Отобразить всех работников.
99             display_workers(workers)
100         elif command.startswith('select '):
101             # Разбить команду на части для выделения стажа.
102             parts = command.split(' ', maxsplit=1)
103             # Получить требуемый стаж.
104             period = int(parts[1])
105             # Выбрать работников с заданным стажем.
106             selected = select_workers(workers, period)
107             # Отобразить выбранных работников.
108             display_workers(selected)
109         elif command == 'help':
110             # Вывести справку о работе с программой.
111             print("Список команд:\n")
112             print("add - добавить работника;")
113             print("list - вывести список работников;")
114             print("select <стаж> - запросить работников со стажем;")
115             print("help - отобразить справку;")
116             print("exit - завершить работу с программой.")
117         else:
118             print(f"Неизвестная команда {command};", file=sys.stderr)
119
120
121 if __name__ == '__main__':
122     main()

```

Рисунок 4 – код программы (конец)

```
>>> add
Фамилия и инициалы? Мамонтов Д.В
Должность? Разработчик
Год поступления? 2020
>>> list
```

No	Ф.И.О.	Должность	Год
1	Мамонтов Д.В	Разработчик	2020

Рисунок 5 – результат работы программы

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

Рисунок 6 – код программы

```
Введите целое число: 3
Число положительное.

Process finished with exit code 0
```

Рисунок 7 – результат работы программы при number = 3

```
Введите целое число: -5
Число отрицательное.

Process finished with exit code 0
```

Рисунок 8 – результат работы программы при number = -5

```
Введите целое число: 0
Число равно нулю.

Process finished with exit code 0
```

Рисунок 9 – результат работы программы при number = 0

```
1  ▶  #!/usr/bin/env python3
2      #- coding: utf-8 -*-
3
4      from math import pi
5
6
7      def cylinder():
8
9          def circle(rad):
10             return pi * rad * rad
11
12         r = int(input("Введите радиус: "))
13         h = int(input("Введите высоту: "))
14         choose = input("Площадь боковой поверхности цилиндра - a\n"
15                        "Полная площадь цилиндра - b\n"
16                        "a/b: ")
17         if choose == 'a':
18             print(f"Площадь боковой поверхности цилиндра = {2 * pi * r * h}")
19         else:
20             print(f"Полная площадь цилиндра = {2 * pi * r * h + 2 * circle(r)}")
21
22
23  ▶  if __name__ == '__main__':
24         cylinder()
```

Рисунок 10 – код программы

```
Введите радиус: 3
Введите высоту: 6
Площадь боковой поверхности цилиндра - a
Полная площадь цилиндра - b
a/b: a
Площадь боковой поверхности цилиндра = 113.09733552923255

Process finished with exit code 0
```

Рисунок 11 – результат работы программы с выбором a

```
Введите радиус: 3
Введите высоту: 6
Площадь боковой поверхности цилиндра - a
Полная площадь цилиндра - b
a/b: b
Полная площадь цилиндра = 169.64600329384882

Process finished with exit code 0
```

Рисунок 12 – результат работы программы с выбором b

```
1 ► 1 #!/usr/bin/env python3
2 2 # -*- coding: utf-8 -*-
3
4 3 def multi():
5     number = int(input("Введите число: "))
6     result = 1
7     if number == 0:
8         return None
9     while number != 0:
10         result *= number
11         number = int(input("Введите число: "))
12     return result
13
14
15 ► 15 if __name__ == '__main__':
16     print(f"Вызов функции и ее результата = {multi()}")
```

Рисунок 13 – код программы

```
Введите число: 0
Вызов функции и ее результата = None

Process finished with exit code 0
```

рисунок 14 – результат работы программы

```
Введите число: 1
Введите число: 2
Введите число: 3
Введите число: 0
Вызов функции и ее результата = 6

Process finished with exit code 0
```

Рисунок 15 – результат работы программы

```
1  ▶  #!/usr/bin/env python3
2      # -*- coding: utf-8 -*-
3
4      def get_input():
5          return input()
6
7
8      def test_input(string):
9          return string.isdigit()
10
11
12     def str_to_int(string):
13         return int(string)
14
15
16     def print_int(integer):
17         print(integer)
18
19
20     def main():
21         data = get_input()
22         if test_input(data):
23             print_int(str_to_int(data))
24
25
26  ▶  if __name__ == '__main__':
27         main()
```

Рисунок 16 – код программы

```
3
3
Process finished with exit code 0
```

Рисунок 17 – результат работы программы

```
sdefrghjkl
Process finished with exit code 0
```

Рисунок 18 – результат работы программы

## ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

Задание:

Решить индивидуальное задание лабораторной работы 2.6, оформив каждую команду в виде отдельной функции.

Код:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import sys

def get_student():
    # Запросить данные о человеке.
    name = input("Фамилия и имя? ")
    phone = input("Номер телефона? ")
    birth = input("Дата рождения? ")
    # Создать словарь.
    return {
        'name': name,
        'phone': phone,
        'birth': birth,
    }

def display_students(staff):
    if staff:
        # Заголовок таблицы.
        line = '+-{}-+-{}-+-{}-+-{}-+'.format(
            '-' * 4,
            '-' * 30,
            '-' * 20,
            '-' * 13
        )
        print(line)
        print(
            '| {:^4} | {:^30} | {:^20} | {:^13} |'.format(
                "No",

```



```

        "Фамилия и имя",
        "Номер телефона",
        "Дата рождения"
    )
)
print(line)
# Вывести данные о всех людях.
for idx, student in enumerate(staff, 1):
    print(
        '| {:>4} | {:<30} | {:<20} | {:>13} |'.format(
            idx,
            student.get('name', ''),
            student.get('phone', ''),
            student.get('birth', '')
        )
    )
print(line)
else:
    print("Список людей пуст")

def select_student(staff, phone):
    # Проверить сведения людей из списка.
    result = ""
    found = False
    for one in staff:
        if one.get('phone', '') == phone:
            result = one.get('name', '')
            found = True
    if not found:
        return "Нет человека с таким номером."
    else:
        return result

def date_key(birth):
    data = birth.split(".")
    return data[2], data[1], data[0]

def main():
    # Список людей.
    students = []
    # Организовать бесконечный цикл запроса команд.
    while True:
        # Запросить команду из терминала.
        command = input(">>> ").lower()
        # Выполнить действие в соответствии с командой.
        if command == 'exit':
            break
        elif command == 'add':
            # Запросить данные о человеке
            student = get_student()
            # Добавить словарь в список
            students.append(student)
            # Отсортировать список в случае необходимости
            if len(students) > 1:
                students.sort(key=lambda item: date_key(item.get('birth',
'')))
        elif command == 'list':
            # Отобразить всех людей
            display_students(students)
        elif command.startswith('phone '):
            # Разбить команду на части для выделения номера телефона.

```

```

        parts = command.split(' ', maxsplit=1)
        # Получить требуемый номер.
        phone = parts[1]
        # Отобразить выбранного человека
        print(select_student(students, phone))
    elif command == 'help':
        # Вывести справку о работе с программой.
        print("Список команд:\n")
        print("add - добавить человека;")
        print("list - вывести список людей;")
        print("phone <номер> - запросить человека по номеру;")
        print("help - отобразить справку;")
        print("exit - завершить работу с программой.")
    else:
        print(f"Неизвестная команда {command}", file=sys.stderr)

if __name__ == '__main__':
    main()

```

```

>>> add
Фамилия и имя? Мамонтов Даниил
Номер телефона? +79624425484
Дата рождения? 03.06.2002
>>> add
Фамилия и имя? Антошкин Алексей
Номер телефона? +79881234567
Дата рождения? 12.12.2001
>>> add
Фамилия и имя? Лазутин Дмитрий
Номер телефона? +79187654321
Дата рождения? 01.07.2002
>>> list
+-----+-----+-----+-----+
| No |          Фамилия и имя          |      Номер телефона      | Дата рождения |
+-----+-----+-----+-----+
|  1 | Антошкин Алексей              | +79881234567             | 12.12.2001 |
|  2 | Мамонтов Даниил              | +79624425484             | 03.06.2002 |
|  3 | Лазутин Дмитрий              | +79187654321             | 01.07.2002 |
+-----+-----+-----+-----+
>>> phone +79624425484
Мамонтов Даниил

```

Рисунок 1 – результат работы программы

## КОНТРОЛЬНЫЕ ВОПРОСЫ

1) Каково назначение функций в языке программирования Python?

Функция 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`. 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?

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. К глобальной переменной можно обратиться из локальной области видимости. К локальной переменной нельзя обратиться из глобальной области видимости, потому что локальная переменная существует только в момент выполнения тела функции.

4) Как вернуть несколько значений из функции Python?

В Питоне позволительно возвращать из функции несколько объектов, перечислив их через запятую после команды `return`.

5) Какие существуют способы передачи значений в функцию?

С помощью так называемых параметров, которые указываются в скобках в заголовке функции. Количество параметров может быть любым.

Однако в Python у функций бывают параметры, которым уже присвоено значение по умолчанию. В таком случае, при вызове можно не передавать соответствующие этим параметрам аргументы. Хотя можно и передать.

6) Как задать значение аргументов функции по умолчанию?

```
def do_something(a, b=2) # Значение по умолчанию b = 2
```

7) Каково назначение `lambda`-выражений в языке Python?

интересный синтаксис, позволяющий определять небольшие однострочные функции на лету. `lambda` – это выражение, а не инструкция. По этой причине ключевое слово `lambda` может появляться там, где синтаксис языка Python не позволяет использовать инструкцию `def`, – внутри литералов или в вызовах функций, например.

8) Как осуществляется документирование кода согласно PEP257?

- Тройные кавычки используются даже если строка помещается на одной линии. Это облегчает последующее расширение документации.
- Закрывающие кавычки находятся на той же строке, что и открывающие. Для однострочных `docstring` это выглядит лучше.

- Ни до, ни после документации не пропускаются строки. Код пишется сразу же на следующей линии
- Документационная строка — это «фраза», заканчивающаяся точкой. Она описывает эффект функции или метода в командном тоне
- Однострочная документация НЕ должна быть простой «подписью», повторяющей параметры функции/метода

Многострочные:

- Многострочные документации состоят из сводной строки (summary line) имеющей такую же структуру, как и однострочный docstring, после которой следует пустая линия, а затем более сложное описание.
- Оставляйте пустую строку после всех документаций (однострочных или многострочных), которые используются в классе;
- Документация скрипта (автономной программы) представляет из себя сообщение «о правильном использовании» и возможно будет напечатано, когда скрипт вызовется с неверными или отсутствующими аргументами
- Документация модуля должна обычно содержать список классов, исключений и функций (и любых других важных объектов), которые экспортируются при помощи библиотеки, а также однострочное пояснение для каждого из них.
- Документация функции или метода должна описывать их поведение, аргументы, возвращаемые значения, побочные эффекты, возникающие исключения и ограничения на то, когда они могут быть вызваны.
- Документация класса должна обобщать его поведение и перечислять открытые методы, а также переменные экземпляра.
- Если класс является потомком и его поведение в основном наследуется от основного класса, в его документации необходимо упомянуть об этом и описать возможные различия.

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

Одиночные строки документации предназначены для действительно очевидных случаев. Они должны уместиться на одной строке.

Многострочные строки документации состоят из однострочной строки документации с последующей пустой строкой, а затем более подробным описанием.