МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования «Северо-Кавказский федеральный университет»

Кафедра инфокоммуникаций

Отчет по лабораторной работе №2.6

по дисциплине «Основы программной инженерии»

Выполнил студент группы ПИЖ-б-о-20-1
Примаков В. Д. « »_____20__г.
Подпись студента_____
Работа защищена « »_____20__г.
Проверил Воронкин Р.А. _____
(подпись)

ВЫПОЛНЕНИЕ

Пример

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

import sys
from datetime import date


def get_worker():
    """
    Запросить данные о работнике.
    """

    name = input("Фамилия и инициалы? ")
    post = input("Должность? ")
    year = int(input("Год поступления? "))

    # Создать словарь.
    return {
        'name': name,
        'post': post,
        'year': year,
    }


def display_workers(staff):
    """
    Отобразить список работников.
    """

    # Проверить, что список работников не пуст.
    if staff:
        # Заголовок таблицы.
        line = '+-{}-+-{}-+-{}-+-{}-+'.format(
            '-' * 4,
            '-' * 30,
            '-' * 20,
            '-' * 8
        )
```

```
        print(line)
        print(
            '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
                "№",
                "Ф.И.О.",
                "Должность",
                "Год"
            )
        )
        print(line)

        # Вывести данные о всех сотрудниках.
        for idx, worker in enumerate(staff, 1):
            print(
                '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
                    idx,
                    worker.get('name', ''),
                    worker.get('post', ''),
                    worker.get('year', 0)
                )
            )
        print(line)

    else:
        print("Список работников пуст.")


def select_workers(staff, period):
    """
    Выбрать работников с заданным стажем.
    """

    # Получить текущую дату.
    today = date.today()

    # Сформировать список работников.
    result = []
    for employee in staff:
        if today.year - employee.get('year', today.year) >= period:
```

```
            result.append(employee)

    # Возвратить список выбранных работников.
    return result


def main():
    """
    Главная функция программы.
    """

    # Список работников.
    workers = []

    # Организовать бесконечный цикл запроса команд.
    while True:
        # Запросить команду из терминала.
        command = input(">>> ").lower()

        # Выполнить действие в соответствие с командой.
        if command == 'exit':
            break

        elif command == 'add':
            # Запросить данные о работнике.
            worker = get_worker()

            # Добавить словарь в список.
            workers.append(worker)
            # Отсортировать список в случае необходимости.
            if len(workers) > 1:
                workers.sort(key=lambda item: item.get('name', ''))

        elif command == 'list':
            # Отобразить всех работников.
            display_workers(workers)

        elif command.startswith('select '):
```

```
            # Разбить команду на части для выделения стажа.
            parts = command.split(' ', maxsplit=1)
            # Получить требуемый стаж.
            period = int(parts[1])

            # Выбрать работников с заданным стажем.
            selected = select_workers(workers, period)
            # Отобразить выбранных работников.
            display_workers(selected)

        elif command == 'help':
            # Вывести справку о работе с программой.
            print("Список команд:\n")
            print("add - добавить работника;")
            print("list - вывести список работников;")
            print("select <стаж> - запросить работников со стажем;")
            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 инициалы? Примаков В.Д.
Должность? Студент
Год поступления? 2020
>>> add
Фамилия и инициалы? Турклиев В.Н.
Должность? Студент
Год поступления? 2020
>>> add
Фамилия и инициалы? Катарин П.В.
Должность? Студент
Год поступления? 2016
>>> add
Фамилия и инициалы? Путин В.В.
Должность? Царь
Год поступления? 1999
>>> list
+------+--------------------------------+----------------------+----------+
|  №   |            Ф.И.О.              |      Должность       |   Год    |
+------+--------------------------------+----------------------+----------+
|    1 | Катарин П.В.                   | Студент              |     2016 |
|    2 | Примаков В.Д.                  | Студент              |     2020 |
|    3 | Путин В.В.                     | Царь                 |     1999 |
|    4 | Турклиев В.Н.                  | Студент              |     2020 |
+------+--------------------------------+----------------------+----------+
>>>
```

```
>>> select 1
   1: Катарин П.В.
   2: Примаков В.Д.
   3: Путин В.В.
   4: Турклиев В.Н.
>>> select 4
   1: Катарин П.В.
   2: Путин В.В.
>>> select 20
   1: Путин В.В.
```

Задание 8. (не важно расположение positive и negative)

8. 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".

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

def positive():
    print("Positive")


def negative():
    print("Negative")


def test():
    a = int(input("Put int num: "))
    if a > 0:
        positive()
    else:
        negative()


# def positive():
#     print("Positive")
#
#
# def negative():
#     print("Negative")


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

```
Put int num: -6
Negative
```

Задание 10.

10. 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 πr^2. В теле *cylinder()* у пользователя спрашивается, хочет ли он получить только площадь боковой поверхности цилиндра, которая вычисляется по формуле $2\pi rh$, или полную площадь цилиндра. В последнем случае к площади боковой поверхности цилиндра должен добавляться удвоенный результат вычислений функции *circle()*.

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

PI = 3.14


def circle(r):
    return PI * (r * r)


def cylinder():
    h = float(input("Put high of cylinder: "))
    r = float(input("Put radius of cylinder: "))
    x = int(input("1 - Calc only side area\n2 - Calc full area\n"))

    if x == 1:
        area = 2 * PI * r * h
    elif x == 2:
        area = 2 * PI * r * h
        area = 2 * (circle(r))
    print(area)


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

```
Put high of cylinder: 5
Put radius of cylinder: 8
1 - Calc only side area
2 - Calc full area
1
251.20000000000002

Process finished with exit code 0
```

```
Put high of cylinder: 5
Put radius of cylinder: 8
1 - Calc only side area
2 - Calc full area
2
401.92

Process finished with exit code 0
```

Задание 12.

12. 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.

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

def mult():
    x = 1
    while True:
        a = int(input("Put numbers you want mult(0 - stop): "))
        if a == 0:
            break
        else:
            x = x * a
    if x == 1:
        exit()
    else:
        print(x)


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

```
Put numbers you want mult(0 - stop): 0

Process finished with exit code 0
```

```
Put numbers you want mult(0 - stop): 6
Put numbers you want mult(0 - stop): 5
Put numbers you want mult(0 - stop): 10
Put numbers you want mult(0 - stop): 0
300

Process finished with exit code 0
```

Задание 12.

14. 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.

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

def get_input():
    a = input("Put some string: ")
    return a


def test_input(a):
    return a.isnumeric()


def str_to_int(a):
    return int(a)


def print_int(a):
    print(a)


if __name__ == '__main__':
    x = get_input()
    if test_input(x) == 1:
        x = str_to_int(x)
        print_int(x)
```

```
Put some string: 7893489
7893489

Process finished with exit code 0
```

```
Put some string: Hello

Process finished with exit code 0
```

Индивидуальное задание

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

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

import sys
from datetime import date


def get_goods():
    """
    Запросить данные о товаре.
    """

    name = input("Название товара: ")
    shop = input("Название магазина: ")
    price = float(input("Стоимость: "))

    # Создать словарь.
    return {
        'name': name,
        'shop': shop,
        'price': price,
    }


def display_goods(goods):
    """
    Отобразить список товаров.
    """
    print(goods)
    # Проверить, что список товаров не пуст.
    if goods:
        # Заголовок таблицы.
        line = '+-{}-+-{}-+-{}-+-{}-+'.format(
            '-' * 4,
            '-' * 30,
            '-' * 20,
            '-' * 8
        )
```

```
        print(line)
        print(
            '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(
                "№",
                "Название",
                "Магазин",
                "Цена"
            )
        )
        print(line)
        # Вывести данные о всех товарах.
        for idx, good in enumerate(goods, 1):
            print(
                '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(
                    idx,
                    good.get('name', ''),
                    good.get('shop', ''),
                    good.get('price', 0)
                )
            )
        print(line)

    else:
        print("Список товаров пуст.")


def select_goods(goods, shop):
    """
    Выбрать товары магазина.
    """

    # Получить текущую дату.
    today = date.today()

    # Счетчик записей.
    count = 0
```

```python
    # Сформировать список товаров.
    result = []

    for good in goods:
        if shop == good.get('shop', shop):
            count += 1
            result.append(good)

    # Проверка на отсутствие товаров или выбранного магазина.
    if count == 0:
        print("Такого магазина не существует либо нет товаров.")
    else:
        # Возвратить список выбранных товаров.
        return result


def main():
    """
    Главная функция программы.
    """

    # Список товаров.
    goods = []

    # Организовать бесконечный цикл запроса команд.
    while True:
        # Запросить команду из терминала.
        command = input(">>> ").lower()

        # Выполнить действие в соответствие с командой.
        if command == 'exit':
            break

        elif command == 'add':
            # Запросить данные о товаре.
            good = get_goods()

            # Добавить словарь в список.
```

```
            goods.append(good)
            # Отсортировать список в случае необходимости.
            if len(goods) > 1:
                goods.sort(key=lambda item: item.get('name', ''))

        elif command == 'list':
            # Отобразить все товары.
            display_goods(goods)

        elif command.startswith('select '):
            # Разбить команду на части для выделения стажа.
            parts = command.split(' ', maxsplit=1)
            # Получить требуемые товары.
            shop = parts[1]

            # Выбрать товары ммагазина.
            selected = select_goods(goods, shop)
            # Отобразить выбранные товары.
            display_goods(selected)

        elif command == 'help':
            # Вывести справку о работе с программой.
            print("Список команд:\n")
            print("add - добавить товар;")
            print("list - вывести список товаров;")
            print("select <имя магазина> - запросить товары магазина;")
            print("help - отобразить справку;")
            print("exit - завершить работу с программой.")
        else:
            print(f"Неизвестная команда {command}", file=sys.stderr)


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

```
>>> help
Список команд:

add - добавить товар;
list - вывести список товаров;
select <имя магазина> - запросить товары магазина;
help - отобразить справку;
exit - завершить работу с программой.
>>>
```

```
Название товара: Сахар
Название магазина: Пятерочка
Стоимость: 55р
>>> add
Название товара: Соль
Название магазина: Пятерочка
Стоимость: 45р
>>> add
Название товара: Печенье
Название магазина: Магнит
Стоимость: 40р
>>> add
Название товара: Торт
Название магазина: Магнит
Стоимость: 250р
>>> list
+------+--------------------------------+----------------------+----------+
|  №   |            Название            |       Магазин        |   Цена   |
+------+--------------------------------+----------------------+----------+
|    1 | Печенье                        | Магнит               |      40р |
|    2 | Торт                           | Магнит               |     250р |
|    3 | Сахар                          | Пятерочка            |      55р |
|    4 | Соль                           | Пятерочка            |      45р |
+------+--------------------------------+----------------------+----------+
>>>
```

```
>>> select пятерочка
пятерочка
   1: Сахар
   2: Соль
>>> select магнит
магнит
   1: Печенье
   2: Торт
>>>
```

```
>>> select перекресток
перекресток
Такого магазина не существует либо нет товаров.
>>>
```

Ссылки на репозитории

GitHub - https://github.com/surai5a/laba_2_8

Ответы на контрольные вопросы

1. Функции решают проблему дублирования кода в разных местах программы. Благодаря им, есть возможность один и тот же участок кода не сразу, а когда понадобится.

2. Функции определяются оператором def. Return возвращает значение, вычисленное функцией в основное тело программы для его дальнейшей обработки.
3. Локальные и глобальные переменные призваны разграничить доступ к переменным между частями кода. Так глобальные переменные доступны в любом месте кода, когда локальные могут быть использованы, например, только в функции, если они были объявлены внутри неё.
4. Вернуть несколько значений сразу можно перечислив их через запятую после оператора return.
5. Передать значение в функцию можно передав имя переменных в качестве параметра, либо передав сами значения переменных.
6. Чтобы задать значение аргументов функции по умолчанию следует после объявлений всех параметров указать те, которые принимают значения по умолчанию в случае, если их значения не будут указаны (def funct(a, b, c = 2)).
7. Лямбды – те же функции, но с упрощенным синтаксисом, и по - сути являются выражениями. Они могут быть использованы там, где не могу функции, внутри литералов или в вызовах функций.
8. Документирование кода по PEP257 предусматривает использование тройных двойных кавычек. Также существует две формы строк документации: однострочная и многострочная.
9. Однострочная строка документации не должна быть "подписью" параметров функции / метода. Этот тип строк документации подходит только для C функций, где интроспекция не представляется возможной. Многострочные строки документации состоят из однострочной строки документации с последующей пустой строкой, а затем более подробным описанием. Первая строка может быть использована автоматическими средствами индексации, поэтому важно, чтобы она находилась на одной строке и была отделена от остальной документации пустой строкой.