

# **Лабораторное занятие 10**

## **Составление рекурсивных функций**

### **1 Цель работы**

1.1 Приобрести навыки по составлению рекурсивных функций.

### **2 Литература**

2.1 Прохоренок, Н.А. Python 3. Самое необходимое / Н.А. Прохоренок, В.А. Дронов. – Санкт-Петербург: БХВ-Петербург, 2016. – с.18-50.

### **3 Подготовка к работе**

3.1 Повторить теоретический материал (см. п.2).

3.2 Изучить описание практической работы.

### **4 Основное оборудование**

4.1 Персональный компьютер.

### **5 Задание**

5.1 Напишите функцию `find_substr(subst, st)`, которая определяет имеет ли строка `st` вхождение `subst` без учета регистра. Если вхождение имеется, то функция должна возвращать слово «True», иначе «False».

Проверки: `print(find_substr('Пит', 'пИтон'))`

```
>>>True
```

```
print(find_substr('программирование', 'ПрограммиРоВаНИЕ'))
```

```
>>>True
```

```
print(find_substr('Довод', 'Повод'))
```

```
>>>False
```

5.2 Напишите функцию `camel(st)`, в которую передается строка `st`, и которая возвращает эту же строку с чередующимися верхним и нижним регистром ее букв. Например, «PyThOn». Первый символ этой строки должен начинаться с верхнего регистра.

Проверки:

```
print(camel('Копейка рубль бережет'))
```

```
>>>КоПеЙкА рУбЛЬ БеРеЖеТ
```

```
print(camel('Из огня да в полымя'))
```

```
>>>Из ОгНя Да В пОлЫмЯ
```

```
print(camel('КРАСОТА'))
```

```
>>>КрАсОтА)))
```

5.3 Написать функцию, вычисляющую факториал числа. Функция должна быть рекурсивной (вызывать сама себя). Перед выполнением ознакомьтесь с приложением.

### **6 Порядок выполнения работы**

6.1 Запустить Python IDLE и выполнить все задания из п.5.

6.2 Ответить на контрольные вопросы.

### **7 Содержание отчета**

- 7.1 Титульный лист
- 7.2 Цель работы
- 7.3 Ответы на контрольные вопросы
- 7.4 Вывод

## 8 Контрольные вопросы

- 8.1 Каким образом можно считать информацию с клавиатуры в приложении на Python?
- 8.2 Какое количество переменных одновременно может быть считано с клавиатуры?
- 8.3 Какой модуль в Python содержит математические функции и константы?
- 8.4 Как преобразовать считанную строку в целочисленный тип данных в Python?

## 9. Приложение

рекурсивная функция — это такая функция, которая в процессе выполнения вызывает саму себя. Это свойство бывает полезно при выполнении некоторых задач в программировании.

### Условия рекурсивных алгоритмов

Вернёмся к нашему примеру с `summa(n)`. Чтобы алгоритм работал, программа должна соответствовать двум требованиям.

#### Базовый случай

Помимо рекурсивного случая, когда функция вызывает сама себя ещё раз, должно быть определённое стоп-условие, чтобы этот процесс не продолжался бесконечно и функция могла завершить работу самостоятельно. В программировании это называется базовым случаем.

У нас он произойдёт, когда `n` станет равным 1. Мы упростили задачу настолько, что больше нет смысла считать, — и можем просто дать ответ.

Чтобы добраться до базового случая, рекурсивной функции приходится вызывать себя определённое количество раз. Такое число самовывозов плюс первоначальный вызов функции называется глубиной рекурсии. В случае `summa(5)` она равна 5.

#### Рекурсия

Чтобы прийти к базовому случаю, мы должны передавать каждой новой функции изменённые данные. Другими словами, нужно изменять аргумент функции.

В нашем коде при первом вызове `n` была равна 5, при втором — 4. И так до тех пор, пока `n` не стала равна 1. Не сделай мы этого, рекурсия `n`

Например, мы хотим написать функцию `summa(n)`, которая считает сумму чисел от 1 до `n`. Если `n = 2`, то результат будет  $1 + 2 = 3$ . Если `n = 5`, то получится  $1 + 2 + 3 + 4 + 5 = 15$ . Реализовать такой алгоритм можно двумя способами: итерационным и рекурсивным.

`summa(5)` — то же самое, что  $5 + \text{summa}(4)$

$\text{summa}(4)$  — то же самое, что  $4 + \text{summa}(3)$

$\text{summa}(3)$  — то же самое, что  $3 + \text{summa}(2)$

$\text{summa}(2)$  — то же самое, что  $2 + \text{summa}(1)$

$\text{summa}(1)$  — это 1

Получается, что мы решаем задачу, используя ответ на эту же задачу, но с меньшей величиной входных данных:

```
def summa(n):  
    if n == 1:  
        return 1  
    return n + summa(n-1)  
  
summa(5)  
>>> 15
```

Схематично работу функции можно обозначить так:

