Функции и работа с файлове в Python

В темата ще бъдат разгледани следните основни въпроси:

- Функция range
- Дефиниране и използване на функции
- Ламбда функция
- Работа с файлове

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

1. Функция range

Функцията range() връща последователност от числа, като се започне от цялото число *start* (0 по подразбиране), числото се увеличава със стъпка *step* и (по подразбиране се увеличава с 1) и спира преди определено число *stop*.

На фиг. 3.8 е представен синтаксиса на функцията range(), която се използва за генериране на числови последователности използвани в обикновено в цикли. Пример за генериране на последователност от 2 до 19 със стъпка 3 е показан на фиг. 3.8.б.

```
range(start, stop, step)
```

а)

```
for n in range(2, 20, 3):
    print(n)
```

б)

Фиг. 3.8. а. Синтаксис на функцията `range`

б., Пример за използване на функцията `range`

Пример, който извежда числа, които не се кратни на 5, в интервала от 10 до 20 е представен на фиг. 3.9.а. Примерът използва операторът continue и функцията range(). Изходът от програмата е показан на фиг. фиг. 3.9.б.

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

```
for num in range(10, 21):
   if num % 5 == 0:
      print ("Multiple of 5: ", num)
      num = num + 1
      continue
   print ("Found number: ", num)
```

а)

```
Multiple of 5:  10
Found number:  11
Found number:  12
Found number:  13
Found number:  14
Multiple of 5:  15
Found number:  16
Found number:  17
Found number:  18
Found number:  19
Multiple of 5:  20
```

б)

Фиг. 3.9. Пример за използване на оператора continue и функцията range в цикъл, който търси и отпечатва числа, които не се делят на 5

а. Код на Python **б.** Изход от програмата

2. Дефиниране и използване на функции

Дефиниция на функция в python започва с ключовата дума def, последвана от името на фукнцията и списък на формалните параметри в малки скоби. Операторите, образуващи тялото на функцията, започват от следващия ред и трябва да бъдат с отстъп навътре. Първият оператор от

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

тялото на функцията може да бъде символен низ, който представлява символен низ за документация на функцията (docstring). Съществуват средства, които използват тези символни низове за документация, за да създадат автоматично печатна документация или позволят лесно разхождане из кода на програмата. Добра практика е да се включват символни низове за документация в кода, но не е задължително да се прави.

Изпълнението на функция въвежда нова символна таблица, в която всички присвоявания на променливи в една функция запазват стойността в локалната символна таблица. В общия случай при обръщения към променливи първо се търси в локалната символна таблица, после в глобалната символна таблица и накрая в таблицата на вградени имена. По този начин, във функцията не може директно да се присвои стойност на глобалните променливи (освен ако не се укаже оператор global), но могат да се правят обръщения към тях.

Дефинирането на функция (фиг. 3.10.а) ще се поясни с преобразуването на кода от фиг. 3.5.а, който търси и отпечатва редицата на Фибоначи. На фиг. 3.10.б е представено извеждането на стринга за документация на функцията и резултатът от извикване на функцията с фактически аргумент 1000.

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

```
def Fib(n):  # Дефиниране на функция Fib с един аргумент n
    '''Изписва числата на Фибоначи до n'''
    a, b = 0, 1
    while b < n:
        print (b, end = ' ')
        a, b = b, a+b

print (Fib.__doc__)
Fib(1000) # Извикаване на дефинираната функция Fib:
```

а)

```
Изписва числата на Фибоначи до n
1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987
```

б)

Фиг. 3.10. Пример за дефиниране на функция, отпечатваща числата на в редицата на Фибоначи до аргумента n

а. Код на Python **б.** Изход от програмата

Актуалните фактически параметри при извикване на функция се въвеждат в локалната символна таблица на извикваната функция, в момента, в който тя бива извикана. Така, аргументите се предават чрез извикване по стойност (където стойността винаги е указател към обект, а не самата стойност на обекта). Ако аргументът е променлив обект, извикващият ще види всички промени, които извикваният е направил по указателя, например вмъкнати елементи в списък.

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

Когато една функция извиква друга функция, се създава нова локална символна таблица за това извикване. Една дефиниция на функция въвежда името на функцията в текущата символна таблица. Стойността на името на функцията има тип, който се разпознава от интерпретатора като дефинирана от потребителя функция. Тази стойност може да бъде присвоявана на друго име, което после също може да бъде използвано като функция. Това служи като общ механизъм за преименуване на функции (фиг. 3.11).

```
f = Fib# Функцията f е преименувана функция на Fib
f(100) # Извикване на функцията f
```

а)

```
1 1 2 3 5 8 13 21 34 55 89
```

б)

Фиг. 3.11. Пример за преименуване на функция Fib, отпечатваща числата на в редицата на Фибоначи до аргумента n

а. Код на Python **б.** Изход от програмата

Функцията Fib не връща стойност. Реално това е стойността None, извеждането на която се подтиска от интерпретатора на Python.

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

Функцията от фиг. 3.10 може лесно да се пренапише като връща списък от числата на Фибоначи, вместо да ги отпечатва. Кодът на таз функция е показан на фиг. 3.12.

```
def FibList(n):
    '''Връща списък, съдържащ числата на Фибоначи до n'''
    result = []
    a, b = 0, 1
    while b < n:
        result.append(b)
        a, b = b, a+b
    return result

f100 = FibList(100) # извиква FibList
print (FibList.__doc__)
print (f100) # Отпечатва резултата
```

а)

```
Връща списък, съдържащ числата на Фибоначи до n
[1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
```

б)

Фиг. 3.12. Пример за дефиниране на функция, връщаща списък, съдържащ числата в редицата на Фибоначи до аргумента n

а. Код на Python **б.** Изход от програмата

Операторът return се използва за прекратяване на функцията и връща стойност в името на функцията. Операторът return без израз като аргумент се използва за връщане от средата на процедура, аналогично на стигането до

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

края й, с което също се излиза от процедурата. В този случай се връща стойността None.

Операторът result.append(b) извиква метод append() на обекта result от тип списък. Методът append() добавя нов елемент в края на списъка. В този пример той е еквивалентен на ‘result = result + [b]’, но е по-ефикасен.

3. Ламбда функция

Програмният език Python позволява да се дефинира малка анонимна функция, която се нарича ламбда функция.

Ламбда функцията може да приеме произволен брой аргументи, но може да има само един израз. На фиг. 3.13.а. е показан синтаксиса за дефиниране на ламбда функция, а на фиг. 3.13.б. е представен пример за дефиниране на ламбда функция с два аргумента `a` и `b`, която пресмята и връща техното произведение `a * b`.

```
lambda arguments : expression
```

а)

```
x = lambda a, b : a * b
print(x(2, 6))
```

б)

Фиг. 3.13. а. Синтаксис на ламбда функция
б., Пример за използване на ламбда функция

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

4. Работа с файлове

Python има няколко функции за създаване, четене, актуализиране и изтриване на файлове.

За да се извърши обработка на файл, трябва да се изпълнят следните три стъпки:

1. Отваряне на файл
2. Четене или запис на файл
3. Затваряне на файла

Python има вградена функция open() за отваряне на файл. Тази функция връща файл обект, наричан още манипулатор. Той се използва съответно за четене или промяна на файла.

Функцията open() приема два параметъра; име на файл и режим на отваряне.

Съществуват четири различни режима за отваряне на файл:

1. **"r" (Четене)** – Това е стойност по подразбиране. Отваря файл за четене. Дава грешка, ако файлът не съществува
2. **"a" (Добавяне)** - Отваря файл за добавяне. Създава файла, ако не съществува
3. **"w" (Записване)** - Отваря файл за запис. Създава файла, ако не съществува
4. **"x" (Създаване)** - Създава посочения файл. Връща грешка, ако файлът съществува

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

Освен това като втори режим може да се посочва, дали файлът да се обработва в двоичен или текстов режим

1. **"t" (Текст)** - Стойност по подразбиране. Текстов режим
2. **"b" (Двоичен)** - Двоичен режим (напр. изображения)

Трябва да внимаваме с режима „w“, тъй като той презаписва файла и ако той вече съществува, всички предишни данни се изтриват.

Записването на низ или последователност от байтове за двоични файлове се извършва с помощта на метода write(). Пример за записване на файл и неговото четене е показан на фиг. 3.14.а. Изходът от програмата е представен на фиг. 3.14.б.

В примера са показани трите основни начина за четене. Това може да стане с метода read(). По подразбиране методът read() връща целия текст на файла, но може да се посочи колко знака от файла да се върнат (примера е посочено 4 знака). За прочитане на цял ред от файла се използва метода readline(). Методът може да има като параметър размер, който, аналогично на read(), посочва колко знака (байта) от реда да се върнат.

След приключване с операциите, с които се обработва файла, той трябва да се затвори с помощта на метода close(). Затварянето на файл ще освободи ресурсите, които са били свързани с файла.

Пример за отваряне на файла File.txt и добавяне на съдържание в края му е показан на фиг. 3.15.а, а изхода от програмата на фиг. 3.15.б.

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

```
f = open("File.txt", "w")
f.write("Hello!\n")
f.write("This is my first file in Phyton \n")
f.close()

file = open("File.txt", "r")
for line in file:
   print (line)
file.close()

f = open("File.txt", "r")
print(f.read())
f.close()

f = open("File.txt", "r")
print(f.read(4))
f.close()

f = open("File.txt", "r")
print(f.readline())
f.close()
```

а)

```
Hello!

This is my first file in Phyton

Hello!
This is my first file in Phyton

Hell
Hello!
```

б)

Фиг. 3.14. Пример за запис на файл и неговото четене

а. Код на Python **б.** Изход от програмата

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

```
f = open("File.txt", "a")
f.write("Now the file content is OK!")
f.close()

f = open("File.txt", "r")
print(f.read())
```

а)

```
Hello!
This is my first file in Phyton
Now the file content is OK!
```

б)

Фиг. 3.15. Пример за отваряне на файла File.txt и добавяне на съдържание в края му **а.** Код на Python **б.** Изход от програмата

За да се изтрие файл, трябва да се използва модула OS и да се изпълни неговата функция os.remove(). Препоръчва се първо да се провери дали файлът съществува. Пример за изтриване на файла File.txt е показан на фиг. 3.16.

```
import os
if os.path.exists("File.txt"):
  os.remove("File.txt")
else:
  print("The file does not exist")
```

Фиг. 3.16. Изтриване на файла File.txt

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.

Аналогично може да се изтрива папка като се използва метода os.rmdir() с аргумент името на папката. За да се изтрие папка, тя трябва да е празна.

Проект BG05M2OP001-2.016-0003„Модернизация на Национален военен университет "В. Левски" - гр. Велико Търново и Софийски университет "Св. Климент Охридски" - гр. София, в професионално направление 5.3 Компютърна и комуникационна техника“, финансиран от Оперативна програма „Наука и образование за интелигентен растеж“, съфинансирана от Европейския съюз чрез Европейските структурни и инвестиционни фондове.