

Практическое занятие №13

Тема: Составление программ в функциональном стиле в IDE PyCharm Community

Цель: закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составление программ с использованием списковых включений, итераторов, генераторов в IDE PyCharm Community.

Постановка задачи №1: Организовать и вывести последовательность из 20 целых чисел, выбрать не повторяющиеся элементы, найти их количество. Элементы больше 5 увеличить в два раза.

Тип алгоритма: смешанный

Текст программы:

```
# Вариант №28
# Организовать и вывести последовательность из 20 целых чисел, выбрать
# не повторяющиеся элементы, найти их количество.
# Элементы больше 5 увеличить в два раза.

from random import randint

# Создаем список с рандомными целыми числами
numbers = [randint(1, 21) for i in range(randint(20, 20))]
length = len(numbers)
print(f'Старый список: {numbers}')

# Находим уникальные элементы и их количество
list_of_unique_numbers = []
for i in numbers:
    if numbers.count(i) == 1:
        list_of_unique_numbers.append(i)
a = len(list_of_unique_numbers)

# Увеличиваем их в два раза (n > 5)
list_of_unique_numbers_new = [n * 5 for n in list_of_unique_numbers if
n > 5]

# Выводим данные
print(f'Не повторяющиеся элементы: {list_of_unique_numbers}')
print(f'Их количество: {a}')
print(f'Элементы больше 5 увеличить в два раза:
{list_of_unique_numbers_new}'')
```

Протокол работы программы:

```
Старый список: [10, 9, 17, 14, 15, 18, 16, 16, 5, 17, 1, 2, 6, 21, 10, 1, 10, 8, 2, 3]
Не повторяющиеся элементы: [9, 14, 15, 18, 5, 6, 21, 8, 3]
Их количество: 9
Элементы больше 5 увеличить в два раза: [45, 70, 75, 90, 30, 105, 40]
Process finished with exit code 0
```

Постановка задачи №2: Составить генератор (yield), который переведет символы строки из верхнего регистра в нижний.

Тип алгоритма: смешанный

Текст программы:

```
# Вариант №28
# Составить генератор (yield), который переведет символы строки из верхнего регистра в нижний.

# Функция, переведет символы строки из верхнего регистра в нижний.
def func_generator(ls):
    for i in ls:
        if type(i) == str:
            i = i.lower()
        yield i

lst = ["CAT", "DOG", "SNAKE"] # входные данные
q = [] # список, в который будет записан результат
generator = func_generator(lst) # создаем генератор

# Добавляем элементы, которые возвращает yield, в список output
for j in generator:
    q.append(j)

print("Входные данные: ", lst)
print("Результат: ", q) # вывод результата
```

Протокол работы программы:

```
Входные данные: ['CAT', 'DOG', 'SNAKE']
Результат: ['cat', 'dog', 'snake']
Process finished with exit code 0
```

Вывод: в процессе выполнении практического занятия закрепил усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрел навыки составление программ с использованием списковых включений, итераторов, генераторов. Выполнена разработка кода,

отладка, тестирование, оптимизация программного кода. Готовые программные коды выложены на GitHub.