

## 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:** в последовательности на  $n$  целых элементов найти среднее арифметическое элементов первой трети

**Тип алгоритма:** смешанный

**Текст программы:**

```
# Вариант 19
# В последовательности на  $n$  целых элементов найти среднее
# арифметическое первой трети.
from random import randint

n = int(input("Укажите длину последовательности (целое число >=
3): ")) # входные данные

# Повторный ввод на случай некорректных входных данных
while n < 3:
    n = int(input("Укажите длину последовательности (целое число
>= 3): "))

lst = [randint(-100, 100) for i in range(n)] # создание списка
из  $n$  случайных целых чисел
average = sum([i for i in lst[:n // 3]]) / (n // 3) # поиск
среднего арифметического в первой трети списка

# Вывод результата
print(f"Последовательность на {n} элементов: ", *lst,
      f"\nСреднее арифметическое первой трети: {average}")
```

**Протокол работы программы:**

Укажите длину последовательности (целое число  $>= 3$ ): 6

Последовательность на 6 элементов: -4 -8 -83 -67 -37 65

Среднее арифметическое первой трети: -6.0

Process finished with exit code 0

**Постановка задачи №2:** составить генератор (yield), который преобразует все буквенные символы в заглавные.

**Тип алгоритма:** смешанный

**Текст программы:**

```
# Вариант 19
# Составить генератор (yield), который преобразует все буквенные
# символы в заглавные

# Функция с ключевым словом yield, которая преобразует все буквенные
# символы в заглавные
def func_generator(input_lst):
    for i in input_lst:
        if type(i) == str:
            i = i.upper()
        yield i

lst = [1, "hello", True, 2, "world!"] # входные данные (список
# элементов с разными типами данных)
output = [] # список output, в который будет записан результат
my_generator = func_generator(lst) # создаем генератор

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

print(output) # вывод результата
```

**Протокол работы программы:**

```
[1, 'HELLO', True, 2, 'WORLD!']
```

```
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.