Практическое занятие.

Тема: Лямбда – функции.

Теоретическая часть.

Лямбда-выражения в языке Python представляют небольшие анонимные функции, которые определяются с помощью оператора lambda. Формальное определение лямбда-выражения:

```
lambda параметры : инструкция
```

Лямбда-функции в Python являются анонимными. Это означает, что функция безымянна. Как известно, ключевое слов def используется в Python для определения обычной функции. В свою очередь, ключевое слово lambda используется для определения анонимной функции**.**

Пример 1.

Возведение числа в степень двойки.

```
>>> degree_two = lambda x: x ** 2
>>> print(degree_two(2))
```

Выходные данные:

```
>>> 4
```

Аналогичным способом можно написать функцию def.

```
>>> def degree_two(x):
>>>>>> x **= 2
>>>>>> return x
>>> print(degree_two(3))
```

Выходные данные:

```
>>> 9
```

Разберём разницу написания двух функций.

Без использования лямбды: Здесь обе функции возвращают заданное значение, возведенное в квадрат. Но при использовании def, нам пришлось определить функцию с именем и degree_two() дать ей входную величину. После выполнения нам также понадобилось возвратить результат, из того места, откуда была вызвана функция, и мы сделали это, используя ключевое слово return.

С применением лямбды: Определение лямбды не включает оператор return, а всегда содержит возвращенное выражение. Мы также можем поместить определение лямбды в любое место, где ожидается функция, и нам не нужно присваивать его переменной. Так выглядят простые лямбда-функции.

Рассмотрим дополнительные функции.

Функция filter()

Функция вызывается со всеми элементами в списке, и в результате возвращается новый список, содержащий элементы, для которых функция результирует в True.

Пример 2.

Существует список, вывести все числа из него кратные 3.

[-3, -4, 1, 6, 8, 9, -10, 10, 12] Назовём его my_list1.

```
>>> my_list1 = [-3,  -4, 1, 6, 8, 9, -10, 10, 12]
>>> new1 = list(filter(lambda n: (n % 3 == 0), my_list1))
>>> print(new1)
```

Выходные данные:

```
>>> 9
```

Функция map()

Функция вызывается со всеми элементами в списке, и в результате возвращается новый список, содержащий элементы, возвращенные данной функцией для каждого исходного элемента.

Пример 3.

Существует список, возвести все его числа в степень 2.

[2, 4, 6, 8, 10, 12, 14, 16, 18, 20] Назовём его my_list1.

```
>>> my_list2 = [2, 4, 6, 8, 10, 12, 14, 16, 18, 20]
>>> new2 = list(map(lambda x: x ** 2, my_list2))
>>> print(new2)
```

Выходные данные:

```
>>> [4, 16, 36, 64, 100, 144, 196, 256, 324, 400]
```

Функция lambda и цикл for.

Лямбда-функция и использование цикла for на примере.

Пример 4.

```
>>> tables = [lambda x = x: x*10 for x in range(1, 5+1)]
>>> for table in tables:
>>>>>> print(table())
```

Выходные данные:

```
>>> 10
>>> 20
>>> 30
>>> 40
>>> 50
```

Условия для lambda.

Рассмотрим использование условий if-else в лямбда-функции. Как вы знаете, Python позволяет нам использовать однострочные условия, и именно их мы можем помещать в лямбда-функцию для обработки возвращаемого результата.

Пример 5.

Выведем наибольшее число.

```
>>> max_number = lambda a, b: a if a > b else b
>>> print(max_number(3, 5))
```

Выходные данные:

```
>>> 5
```

На основании теоретических данных, выполните следующие задания в практической части ниже.

Практическая часть.

Задание 1.

Создайте функцию умножения двух чисел, с помощью lambda.

Входные данные:

>>> print(x(2, 3))

Выходные данные:

>>> 6

Задание 2.

Пользователь сначала вводит количество чисел, а затем сами числа. После чего они добавляются в список.

filter проверяет числа в списке кратные 3 и 5 и выводит на экран.

Алгоритм выполнения.

1. Создание переменной для ввода чисел.

2. Создание списка.

3. Добавление цикла for, в диапазоне от нуля до максимального количества чисел.

4. Реализация переменной для ввода чисел.

5. Добавление числе в список.

6. Реализация функции filter, для списка. В данном случае числа lambda должны быть кратные 3 и 5.

7. Вывод lambda.

Входные данные:

>>> Всего чисел будет: 3

>>> 15

>>> 10

>>> 3

Выходные данные:

>>> [15]

Критерии выставления оценок.

Оценка 5 – Полностью выполнено 2 задачи.

Оценка 4 - Первая задача написана полностью, однако вторая программа, имеет ошибки, препятствующие выполнению программы.

Оценка 3 – Решена только первая задача.

Оценка 2 – Работа не сдана.