

2.2. Задача, формирование таблицы значений функции

Задача. Построить таблицу значений функции $f(x)$ на интервале $[a, b]$, количество значений в таблице n .

$$f(x) = \frac{1}{x+1} - \frac{x}{x-3}.$$

2.2.1. Описание алгоритма

Алгоритм построения таблицы значений функции на интервале $[a, b]$, количество значений n :

- вычислить шаг h по формуле:

$$h = \frac{b-a}{n-1};$$

- создать список **x_list**, содержащий все значения аргумента функции **x**, список должен включать следующие значения:

$$a, a+h, a+2 \cdot h, a+3 \cdot h, \dots, b;$$

- создать список **y_list**, содержащий все значения функции, вычисленные от сформированных в списке аргументов, список должен включать:

$$f(x_{list[0]}), f(x_{list[1]}), \dots, f(x_{list[n-1]});$$

- вывести таблицу значений функции, в первом столбце которой будут располагаться элементы списка **x_list**, во втором – соответствующие значения списка **y_list**.

2.2.2. Реализация

Для построения таблицы значений функции используется следующая программа, пояснения к операторам даны в комментариях:

```
import math
# Опишем функцию f(x):
def f_x(x):
    try:
        y = 1 / (x+1) + x / (x-3)
    except:
        y = math.inf
    return y

# Введем границы интервала построения a, b
# и количество значений на интервале n:
a = float(input("a = "))
b = float(input("b = "))
n = int(input("n = "))

# Выполним проверку правильности ввода:
# количество значений должно быть больше 0,
# нижняя граница должна быть меньше верхней
if n < 0 or a >= b:
    print("Ошибочные входные данные")

# Если все введено верно
else:
    # вычислим шаг изменения аргумента функции:
    h = (b - a) / (n - 1)

    # создадим список значений
    x_list = [a + i * h for i in range(n)]

    # вычислим значение функции для каждого значения из списка
    f_list = [f_x(x) for x in x_list]

    # Выведем таблицу значений функции:
    for i in range(n):
        print ("%4.1f : %6.3f" % (x_list[i], f_list[i]))
```

Результат:

```
a = -5
b = 5
n = 21
-5.0 : 0.375
-4.5 : 0.314
-4.0 : 0.238
-3.5 : 0.138
-3.0 : 0.000
-2.5 : -0.212
-2.0 : -0.600
-1.5 : -1.667
-1.0 : inf
-0.5 : 2.143
0.0 : 1.000
0.5 : 0.467
1.0 : 0.000
1.5 : -0.600
2.0 : -1.667
2.5 : -4.714
3.0 : inf
3.5 : 7.222
4.0 : 4.200
4.5 : 3.182
5.0 : 2.667
```

Оформим вывод в виде привычной таблицы (знак * при форматном выводе означает, что строку "-" нужно повторить указанное количество раз). Для этого заменим вывод таблицы в программе на следующий код:

```
# вывод шапки таблицы
print("-" * 17)
print (" | %4s | %6s |" % ("x", "f(x)"))
print("-" * 17)
# вывод содержимого таблицы
for i in range(n):
    print (" | %4.1f | %6.3f |" % (x_list[i], f_list[i]))
# вывод подчеркивания
print("-" * 17)
```

Результат:

```
a = -5
b = 5
n = 21

-----
|      x      |      f(x)      |
-----
| -5.0 | 0.375 |
| -4.5 | 0.314 |
| -4.0 | 0.238 |
| -3.5 | 0.138 |
| -3.0 | 0.000 |
| -2.5 | -0.212 |
| -2.0 | -0.600 |
| -1.5 | -1.667 |
| -1.0 | inf |
| -0.5 | 2.143 |
| 0.0 | 1.000 |
| 0.5 | 0.467 |
| 1.0 | 0.000 |
| 1.5 | -0.600 |
| 2.0 | -1.667 |
| 2.5 | -4.714 |
| 3.0 | inf |
| 3.5 | 7.222 |
| 4.0 | 4.200 |
| 4.5 | 3.182 |
| 5.0 | 2.667 |
-----
```

Задание для самопроверки

Написать программу для решения следующей задачи:

Для вычисления и прогноза численности населения Земли С.П. Капица предложил следующую формулу:

$$N(t) = \frac{C}{\tau} \cdot \operatorname{arcctg} \left(\frac{T_1 - t}{\tau} \right),$$

где t – год, для которого вычисляется численность населения;

C – 172 миллиарда человек/лет;

T_1 – 2000 год;

τ – 45 лет.

Вычислить численность населения в заданные годы.

Реализовать задачу на основе шаблона.

```
import math

def compute_population(t):
    #вычислить численность населения для года t по формуле

#ввести строку с перечисленными через пробел годами
line = input("Введите список лет через пробел: ")

# преобразовать строку в список из строковых значений годов
years_str_list = line.split()

# вычислить количество элементов в списке
n = len(years_str_list)

# сформировать список years_list на основе years_str_list,
#преобразовав строковые значения в целые

# создать список population_list, каждый элемент которого вычисляется
# функцией compute_population от соответствующего года из years_list

# в цикле для каждого года вывести численность населения,
# для вывода использовать формат "%5d - %6.3f миллиард(ов) "
```

Результат

```
Введите список лет через пробел: 1220 2369 2636 1032 2657 198 1118
1220 - 0.220 миллиард(ов)
2369 - 11.544 миллиард(ов)
2636 - 11.738 миллиард(ов)
1032 - 0.178 миллиард(ов)
2657 - 11.746 миллиард(ов)
198 - 0.095 миллиард(ов)
1118 - 0.195 миллиард(ов)
```

Пояснение

1. В модуле **math** функции **arcctg()** нет, но ее можно выразить через **arctg()** следующим образом:

$$\operatorname{arcctg}(x) = \frac{\pi}{2} - \operatorname{arctg}(x).$$

2. Для вычисления **arctg()** в модуле **math** используется **atan()**.