

3 Функции и ветвления

Эта часть знакомит с двумя фундаментальными и крайне полезными понятиями в программировании: определяемые пользователем функции и ветвление потока выполнения программы.

Задания

Задание 3.15 Аппроксимация функции суммой синусов

Рассмотрим кусочно-постоянную функцию

$$f(t) = \begin{cases} 1, & 0 < t < T/2, \\ 0, & t = T/2, \\ -1, & T/2 < t < T \end{cases}$$

Эту функцию можно аппроксимировать суммой

$$S(t, n) = \frac{4}{\pi} \sum_{i=1}^n \frac{1}{2i-1} \sin\left(\frac{2(2i-1)\pi t}{T}\right).$$

Можно показать, что $S(t, n) \rightarrow f(t)$ при $n \rightarrow \infty$.

- а) Напишите функцию `S(t, n, T)`, которая возвращает значение $S(t, n)$.
- б) Напишите функцию `f(t, T)` для вычисления $f(t)$.
- в) Выведите таблицу с информацией, показывающей, как ошибка $f(t) - S(t, n)$ зависит от n и t для случаев $n = 1, 3, 5, 10, 30, 100$ и $t = \alpha T$ при $T = 2\pi$ и $\alpha = 0.01, 0.25, 0.49$.

Примечание Сумма синусов или косинусов как в этом случае называется разложением в ряд Фурье. Аппроксимация функции рядом Фурье — это очень важный прием в науке.

Задание 3.23 Нахождение максимального и минимального значения в списке

Для заданного списка `a` функция `max` из стандартной библиотеки Python вычисляет самый большой элемент в `a`: `max(a)`. Аналогично, `min(a)` возвращает наименьший элемент в `a`. Напишите свои собственные реализации функций `max` и `min`.

Совет. Инициализируйте переменную `max_elem` первым элементом списка, затем пройдите все остальные элементы из `a[1:]`, сравнивая каждый с `max_elem`. Если элемент больше `max_elem`, присвойте значение этого элемента переменной `max_elem`. Используйте аналогичный прием для нахождения минимального элемента.

Задание 3.23 Реализация функции Хевисайда

Следующая функция известна как функция Хевисайда или единичная ступенчатая функция. Эта функция часто используется в математике:

$$H(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

- а) Реализуйте функцию $H(x)$ на Python.
- б) Напишите функцию `test_H()` для тестирования реализации $H(x)$. В этой функции вычислите $H(-10)$, $H(-10^{-15})$, $H(0)$, $H(10^{-15})$, $H(10)$ и проверьте корректность ответов.

Задание 3.26 Реализация кусочно-постоянной функции

Кусочно-постоянные функции имеют много важных применений при моделировании физических явлений с помощью математики. Кусочно-постоянная функция может быть определена как

$$f(x) = \begin{cases} v_0, & x \in [x_0, x_1), \\ v_1, & x \in [x_1, x_2), \\ \vdots & \\ v_i, & x \in [x_i, x_{i+1}), \\ \vdots & \\ v_n, & x \in [x_n, x_{n+1}) \end{cases}$$

Это означает, мы имеем объединение неперекрывающихся интервалов покрывающих область $[x_0, x_{n+1}]$, и $f(x)$ постоянна на каждом интервале.

Один из примеров функции — это -1 на $[0, 1]$, 0 на $[1, 1.5]$ и 4 на $[1.5, 2]$, где в соответствии с определением выше $x_0 = 0$, $x_1 = 1$, $x_2 = 1.5$, $x_3 = 2$ и $v_0 = -1$, $v_1 = 0$, $v_3 = 4$.

- а) Напишите функцию `piecewise(x, data)` для вычисления кусочно-постоянной функции в точке x . Объект `data` — это список пар v_i, x_i при $i = 0, \dots, n$. Например, `data` содержит $[(0, -1), (1, 0), (1.5, 4)]$ для примера, приведенного выше. Т.к. x_{n+1} не является частью (не содержится в) `data`, у нас нет способа определения находится ли x справа от последнего интервала $[x_n, x_{n+1}]$, т.е. мы должны считать, что пользователь использует значения $x \leq x_{n+1}$.
- б) Придумайте подходящие случаи для тестирования функции `piecewise` и реализуйте их в тестовой функции `test_piecewise()`.

Задание 3.35 Подсчет количества вхождений пары символов в строке

Напишите функцию `count_pairs(dna, pair)`, которая возвращает количество появлений пары символов (`pair`) в строке, содержащей код ДНК (`dna`). Например, вызов функции с переменными `dna`, содержащей 'ACTGCTATCCATT' и `pair` — 'AT' должен давать число 2.

Совет. Для каждого совпадения первого символа `pair` и символа строки `dna` проверьте следующий символ в главной строке. Используйте срезы наподобие `s[3:9]`, для выбора части строки `s`.