

Практическая работа 9

Рекурсия

Рекурсия – фундаментальное понятие в математике и компьютерных науках. В языках программирования рекурсивной программой называется программа, которая обращается сама к себе

(подобно тому, как в математике рекурсивная функция определяется через понятия самой этой

функции). Рекурсивная программа не может вызывать себя до бесконечности, следовательно, вторая важная особенность рекурсивной программы – наличие условия завершения, позволяющее

программе прекратить вызывать себя.

В первую очередь надо понимать, что рекурсия — это своего рода перебор. Вообще говоря, всё то, что решается итеративно можно решить рекурсивно, то есть с использованием рекурсивной функции.

Так же, как и у перебора (цикла) у рекурсии должно быть условие остановки — Базовый случай (иначе также как и цикл рекурсия будет работать вечно — infinite). Это условие и является тем случаем, к которому рекурсия идет (шаг рекурсии). При каждом шаге вызывается рекурсивная функция до тех пор, пока при следующем вызове не сработает базовое условие и произойдет остановка рекурсии (а точнее возврат к последнему вызову функции). Всё решение сводится к решению базового случая. В случае, когда рекурсивная функция вызывается для решения сложной задачи (не базового случая) выполняется некоторое количество рекурсивных вызовов или шагов, с целью сведения задачи к более простой. И так до тех пор, пока не получим базовое решение.

Итак, рекурсивная функция состоит из

- Условие остановки или же Базовый случай

Пример подсчет факториала

Шаг рекурсии — способ сведения задачи к более простым.

```
def fac(n):  
    if n == 0:  
        return 1  
    return fac(n-1) * n  
  
print(fac(5))
```

1. шаг. Вызов функции: fac(5)
2. fac(5) возвращает fac(4) * 5
3. fac(4) => fac(3) * 4
4. fac(3) => fac(2) * 3
5. fac(2) => fac(1) * 2
6. fac(1) => 1
7. 1 * 2 - возврат в вызов fac(2)
8. 2 * 3 - fac(3)
9. 6 * 4 - fac(4)
10. 24 * 5 - fac(5)
11. Возврат в основную ветку программы значения 120

Задачи (по 1 заданию из каждого блока) с помощью рекурсии(без циклов) на github загрузить скриншоты вывода программы.

Блок А (ваш 8=1,9 =2,10=3...):

1. Дано натуральные числа X, N Вычислить выражение вида: $x^n / n!$.
2. Дано натуральные числа a, b Вычислить остаток от деления a на b
3. Вывести число в обратном порядке
4. Дано натуральное число N . Вычислите сумму его цифр. При решении этой задачи нельзя использовать строки, списки, массивы
5. Дано натуральное число N . Выведите все его цифры по одной, в обратном порядке, разделяя их пробелами или новыми строками. При решении этой задачи нельзя использовать строки, списки, массивы. Разрешена только рекурсия и целочисленная арифметика.
6. Дано натуральное число $n > 1$. Проверьте, является ли оно простым. Программа должна вывести слово `YES`, если число простое и `NO`, если число составное.
7. Даны два целых числа A и B (каждое в отдельной строке). Выведите все числа от A до B включительно, в порядке возрастания, если $A < B$, или в порядке убывания в противном случае.

Блок Б(с 5-10 вариант

,11...вариант/4 -1,

1. Вводим последовательность натуральных чисел (одно число в строке), завершающаяся числом 0. Определите наибольшее значение числа в этой последовательности. В этой задаче нельзя использовать глобальные переменные и передавать какие-либо параметры в рекурсивную функцию. Функция получает данные, считывая их с клавиатуры. Функция возвращает единственное значение: максимум считанной последовательности. Гарантируется, что последовательность содержит хотя бы одно число (кроме нуля).
2. Дана последовательность натуральных чисел (одно число в строке), завершающаяся числом 0. Определите значение второго по величине элемента в этой последовательности, то есть элемента, который будет наибольшим, если из последовательности удалить наибольший элемент.
3. Дана последовательность натуральных чисел (одно число в строке), завершающаяся числом 0. Выведите первое, третье, пятое и т.д. из введенных чисел. Завершающий ноль выводить не надо. В этой задаче нельзя использовать глобальные переменные и передавать какие-либо параметры в рекурсивную функцию. Функция получает данные, считывая их с клавиатуры. Функция не возвращает значение, а сразу же выводит результат на экран. Основная программа должна состоять только из вызова этой функции.

4. Дано натуральное число $n > 1$. Проверьте, является ли оно простым. Программа должна вывести слово YES, если число простое и NO, если число составное. Алгоритм должен иметь сложность $O(\log n)$.
Указание. Понятно, что задача сама по себе нерекурсивна, т.к. проверка числа n на простоту никак не сводится к проверке на простоту меньших чисел. Поэтому нужно сделать еще один параметр рекурсии: делитель числа, и именно по этому параметру и делать рекурсию.