Лабораторна робота № 7.2

Рекурсивні функцій, системи числення

Мета роботи: Оволодіння практичними навичками у використанні рекурсивних функцій та засвоїти правила переводи чисел між системами числення

Позиційні системи числення

Позиційні системи числення - це системи числення, в яких значення цифри безпосередньо залежить від її положення в числі.
Наприклад, число 01 позначає одиницю, 10 - десять. Кількість цифр, що використовуються в системі числення, називається її «основою». У десятковій системі основа дорівнює десяти, в двійковій системі - двом, ну а в восьмеричній і шістнадцятирічній - відповідно, восьми і шістнадцяти. Тобто в *p*-ічной системі числення кількість цифр одно *p* і використовуються цифри від 0 до *p*-1.

У загальному випадку в позиційній системі числення числа представляються наступним чином: $a_{n-1}...a_1a_0{}_f$, де $a_0, a_1, ..., a_{n-1}$— цифри, а f— основа системи числення. Якщо використовується десяткова система, то f— можна опустить.

Перевід чисел між системами числення

Таке представлення чисел означає ось таке число:

$$a_{n-1}f^{n-1} + ... + a_1f^1 + a_0f^0,$$

де $a_0, a_1, ..., a_{n-1}$— цифри, а f— основа системи числення.

Використовуємо тільки що наведену формулу:

$$25_{10} \to 2 \cdot 10^1 + 5 \cdot 10^0 = 2 \cdot 10 + 5 \cdot 1 = 25_{10}$$

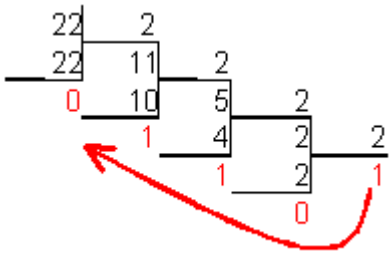
$$31_8 \to 3 \cdot 8^1 + 1 \cdot 8^0 = 3 \cdot 8 + 1 \cdot 1 = 25_{10}$$

$$221_3 \to 2 \cdot 3^2 + 2 \cdot 3^1 + 1 \cdot 3^0 = 2 \cdot 9 + 2 \cdot 3 + 1 \cdot 1 = 25_{10}$$

$$11001_2 \to 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 1 \cdot 16 + 1 \cdot 8 + 0 \cdot 4 + 0 \cdot 2 + 1 \cdot 1 = 25_{10}$$

1. Для переводу десяткового числа в двійкову систему його необхідно послідовно ділити на 2 до тих пір, поки не залишиться залишок, менший або рівний 1. Число у двійковій системі записується як послідовність останнього результату ділення і залишків від ділення в зворотному порядку.

Приклад. Число 22_{10} перевести в двійкову систему числення.



$22_{10} = 10110_2$.

2. Для переводу двійкового числа в десяткове необхідно його записати у вигляді многочлена, що складається з творів цифр числа і відповідного ступеня числа 2, і обчислити за правилами десяткової арифметики

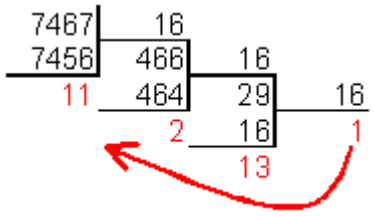
$$X_2 = A_n \cdot 2^{n-1} + A_{n-1} \cdot 2^{n-2} + A_{n-2} \cdot 2^{n-3} + \ldots + A_2 \cdot 2^1 + A_1 \cdot 2^0$$

Приклад. Число 11101000_2 перевести в десяткову систему числення.

$$11101000_2 = 1 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0 = 232_{10}$$

3. Для переводу десяткового числа в шістнадцятирічну систему його необхідно послідовно ділити на 16 доти, поки не залишиться залишок, менший або рівний 15. Число у шістнадцятирічній системі записується як послідовність цифр останнього результату ділення і залишків від ділення в зворотному порядку**.** Для запису чисел в цій системі окрім 10 арабських цифр (від *0* до *9*) використовують 6 літер латинської абетки: 10 - *A*, 11 - *B*, 12 - *C*, 13 - *D*, 14 - *E*, 15 - *F*.

Приклад. Число 7467_{10} перевести в шістнадцятирічну систему числення



$7467_{10} = 1D2B_{16}$

4. Для перекладу шістнадцятирічного числа на десяткове необхідно його записати у вигляді многочлена, що складається з творів цифр числа і відповідного ступеня числа 16, і обчислити за правилами десяткової арифметики**:**

$$X_{16} = A_n \cdot 16^{n-1} + A_{n-1} \cdot 16^{n-2} + A_{n-2} \cdot 16^{n-3} + \ldots + A_2 \cdot 16^1 + A_1 \cdot 16^0$$

Приклад . Число $FDA1_{16}$ перевести в десяткову систему числення.

$$FDA1_{16} = 15 \cdot 16^3 + 13 \cdot 16^2 + 10 \cdot 16^1 + 1 \cdot 16^0 = 64929_{10}$$

Рекурсія

В інженерній практиці приходиться іноді програмувати рекурсивні алгоритми. Така необхідність виникає при реалізації динамічних структур даних, таких як стеки, дерева, черги. Для реалізації рекурсивних алгоритмів передбачена можливість створення рекурсивних функцій. Рекурсивна функція являє собою функцію, у тілі якої здійснюється виклик цієї ж функції.

Використання рекурсивної функції для обчислення факторіала

Нехай потрібно скласти програму для обчислення факторіала будь-якого додатного числа.

```
def fact(x):
    if x == 0:
        return 1
    else:
        return x * fact(x - 1)
```

Для від'ємного аргументу факторіала не існує. Позаяк факторіал 0 дорівнює 1 за означенням, то в тілі функції передбачений і цей варіант. У випадку коли аргумент функції **`fact()`** відмінний від 0, то викликаємо функцію **`fact()`** із зменшеним на одиницю значенням параметра і результат множиться на значення поточного параметра. Таким чином, у результаті вбудованих викликів функцій повертається наступний результат:

```
n *  (n-1)  *  (n-2)   * .   .   .  * 2 * 1 * 1
```

Остання одиниця при формуванні результату належить виклику **`fact(0)`**.

Використання рекурсивної функції для реалізації алгоритму бінарного пошуку у відсортованому масиві

```
def binSearch(arr, value, low, high):
 res = (-1, value)
 if high < low:
      return res
 mid = (low+high)>>1 #швидке ділення на 2 за допомогою
оператора бінарного зсуву, аналог (low+high)/2

 if arr[mid] > value:
      return binSearch(arr, value, low, mid-1)
 elif arr[mid] < value:
      return binSearch(arr, value, mid+1, high)
 else:
      return (mid, value)

arr = [1, 2, 3, 4, 11, 19, 20, 122, 130, 131]
x = 122
print binSearch(arr, x, 0, len(arr))
```

Порядок виконання роботи

Напишіть рекурсивні реалізації функції для вирішення наступних завдань за варіантом. Користуватись циклами не дозволяється.

1. В теории вычислимости важную роль играет функция Аккермана *A*(*m*,*n*), определенная следующим образом:

$$A(m,\ n) = \begin{cases} n+1, & m = 0; \\ A(m-1,\ 1), & m > 0,\ n = 0; \\ A(m-1,\ A(m,\ n-1)), & m > 0,\ n > 0. \end{cases}$$

Напишите реализацию данной функции. Подсчитаетй, значение данной функции при следующем ее вызове *A*(*10*,*5*)

2. Количество разбиений числа *n* на слагаемые, не превышающие k, удовлетворяет рекуррентной формуле:

$$P(n,k) = \begin{cases} P(n,k-1) + P(n-k,k), & k \leqslant n, \\ P(n,n), & k > n, \end{cases}$$

с начальными значениями: $P(0,0) = 1, P(i,0) = 0$ для всех $i > 0$

Напишите реализацию данной функции. Подсчитайте, значение данной функции при следующем ее вызове *P*(*7, 30*)

3. Числа Катала́на — числовая последовательность, встречающаяся во многих задачах комбинаторики. Последовательность названа в честь бельгийского математика Каталана. Числа задются следующим соотношением

$$C_0 = 1 \quad \text{and} \quad C_{n+1} = \sum_{i=0}^{n} C_i\, C_{n-i} \quad \text{for } n \geq 0;$$

Напишите функцию вычисляюще число Каталена. Подсчитайте, значение числа C_n, при n=10

4. Числа Деланноя D(a, b) в комбинаторике описывают количества путей из левого нижнего угла прямоугольной решётки (a, b) в противоположный по диагонали угол, используя только ходы вверх, вправо или вверх-вправо («ходом короля»).

$$D(m,n) = \begin{cases} 1 & \text{if } m = 0 \text{ or } n = 0 \\ D(m-1,n) + D(m-1,n-1) + D(m,n-1) & \text{else} \end{cases}$$

Напишите функцию вычисляюще число Деланноя Подсчитайте, значение числа D(7, 5)

5. Дано натуральное число N. Выведите слово YES, если число N является точной степенью двойки, или слово NO в противном случае.

Операциии возведения в степень, циклы и списки использовать нельзя

6. Дано натуральное число N. Вычислите сумму его цифр.

При решении этой задачи нельзя использовать строки, списки, и циклы.

7. 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, циклы. Разрешена только рекурсия и целочисленная арифметика.

8. Дано натуральное число $n>1$. Проверьте, является ли оно простым. Программа должна вывести слово `YES`, если число простое и `NO`, если число составное.

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.

9. Дано натуральное число $n>1$. Выведите все простые множители этого числа в порядке неубывания с учетом кратности. Пример для 18: 2 3 3

10. Определить, является ли заданная строка палиндромом, т.е. читается одинаково слева направо и справа налево. Выведите YES или NO.

При решении этой задачи нельзя пользоваться циклами. Палиндром — это последовательность символов, которая слева-направо и справа-налево пишется одинаково. Например «АБА» или «АББ ББА».

11. Напишите рекурсивную функцию перемножающую два натуральных числа, не используя операции умножения.

12. Напишите рекурсивную функцию переводящую число в двоичную систему счисления

13. Напишите рекурсивную функцию, проверяющую правильность расстановки скобок в строке. При правильной расстановке выполняются условия:

(а) количество открывающих и закрывающих скобок равно.
(б) внутри любой пары открывающая – соответствующая закрывающая скобка, скобки расставлены правильно.

Примеры неправильной расстановки:)(, ())(, ())(() и т.п.

14. Создайте рекурсивную функцию, подсчитывающую сумму элементов массива (списка) по следующему алгоритму: массив делится пополам, подсчитываются и складываются суммы элементов в каждой половине. Сумма элементов в половине массива подсчитывается по тому же алгоритму, то есть снова путем деления пополам. Деления происходят, пока в получившихся кусках массива не окажется по одному элементу и вычисление суммы, соответственно, не станет тривиальным.

15. Написать рекурсивную функцию сложения двух чисел, используя только прибавление единицы.

16. Написать функцию C(m,n) вычисления биномиальных коэффициентов C_n^m по следующей формуле:

$$C_n^0 = C_n^n = 1; \quad C_n^m = C_{n-1}^m + C_{n-1}^{m-1} \text{ при } 0 < m < n,$$

17. Напишите рекурсивную функцию переводящую число в шестнадцатиричную систему счисления