УПРАЖНЕНИЕ №5

РЕКУРСИВНИ ФУНКЦИИ

Рекурсията е фундаментално понятие в много науки и е от най-мощните средства в програмирането. Думата рекурсия от латински “recursus” означава връщане.

По общо тя представлява метод за дефиниране на алгоритми, понятия, множества и други чрез самите тях. Обикновено при дефиниране на рекурсивни обекти се дефинират един или няколко базови случая, към които след краен брой стъпки се свеждат всички останали случаи.

Рекурсивните алгоритми са изключително удобни при обработка на структури от данни като динамични списъци, дървета, графи и др., както и при алгоритми, чиято структура е рекурсивна.

Рекурсията е особен начин на само повторение. Една функция е рекурсивна, ако в дефиницията й, се съдържа обръщение към самата нея. Такава рекурсия се нарича “пряка”.

Ако една функция (P) се обръща към друга функция (Q), а тя от своя страна се обръща към (P), се казва, че съществува непряка рекурсия.

Примери за рекурсивно дефинирани функции:

Зад1. Факториел:
F(n)=1, ako n=0
F(n)=n*F(n-1), ako n>0

```
long fakt (int n)
{
if  (n <= 1)  return 1;
        else return n * fakt (n - 1);
}
```

Зад2. Числа на Фибоначи:
Fib (0)=0,
Fib (1)=1,
Fib (n)=Fib (n-1)+Fib (n-2), n>1.

Рекурсивната функция за определяне на n-тото число на Фибоначи има следната дефиниция:

```
int Fib(int n)
{    if (n==0) return 0;
     if (n==1)  return 1;
```

```
        else
          return Fib(n-1)+Fib(n-2);
}
```

Да се напише програма, която намира сумата на числата на Фибоначи от интервала [a, b] (a и b са дадени естествени числа, a<b).

Зад3. Проверете дали са в сила релациите:
а) fib(1) + fib(2) + … + fib(n) = fib(n+2)-1;
б) fib(1) + fib(3) + … + fib(2n-1) = fib(2n);
в) $fib^2(1) + fib^2(2) + … + fib^2(n) = fib(n).fib(n+1)$
за всяко цяло число n от интервала [1, 100].

Зад4. Повдигане на степен.
Да се напише рекурсивна функция за повдигане на степен **Step (x, n),** където **x** е реално число **(x≠0**), а **n** е цяло число, която намира $\mathbf{x^n}$, съгласно формулата:

$$\mathbf{x.x^{n-1}}\ \mathbf{, n>0}$$
$$\mathbf{1}\ \mathbf{, n=0}$$
$$\mathbf{1/\ x^{-n}}\ \mathbf{, n<0}$$

Зад5. Намиране на най-голям общ делител на две числа.
Да се намери най-големият общ делител на две цели положителни числа, които се делят на него без остатък.
Рекурсивната реализация на алгоритъма на Евклид, за намиране на най-големият общ делител на две цели положителни числа, се представя със следната рекурсивна функция:

```
int Ngod(int m,  int n)
{
     if    (n == 1) return  m;
     else  return  Ngod(n, m%n);
}
```

Зад6. Да се напише рекурсивна функция, която намира стойността на функцията на Акерман Ack (m, n), дефинирана за $m \geq 0$ и $n \geq 0$ по следния начин:

$$\mathbf{Ack(0, n) = n+1}$$
$$\mathbf{Ack(m, 0) = Ack(m-1, 1),\ m > 0}$$
$$\mathbf{Ack(m, n) = Ack(m-1, Ack(m, n-1)),\ m > 0,\ n > 0.}$$

Зад7. Да се напише рекурсивна функция, която установява, дали в записа на естественото числа n се съдържа цифрата k.

Зад8. За Ханойските кули.

Дадени са три пилона A, B и C. На пилон A са подредени N дискове (N>2) с различен размер, като всеки диск е разположен върху диск с по-голям диаметър. Образува се кула. откъдето идва и името на задачата. Да се опише алгоритъм за прехвърляне на всички дискове от пилон A на пилон C, използвайки за междинно прехвърляне пилон B. На всяка стъпка от един пилон на друг се премества само един диск. Забранява се поставянето на диск с по-голям диаметър върху такъв с по-малък и ли оставяне на диск отстрани на пилоните.

1. При всеки ход може да бъде преместен един диск и този диск трябва да бъде най-горен за някой от стълбовете.

2. Диск с по-голям диаметър не може да бъде поставен върху такъв с по-малък.