Лабораторная работа № 2
Рекурсия.

Задание
В соответствии со своим вариантом n, где n– порядковый номер студента в списке составить программу, реализующая рекурсивную(ые) функцию(и).
Вариант

1. Вычислить несколько значений функции Аккермана для неотрицательных чисел m и n:

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

2. Найти первые N чисел Фибоначчи. Каждое число Фибоначчи равно сумме двух предыдущих чисел при условии, что первые два равны (1, 1, 2, 3, 5, 8, 13, 21,…), поэтому в общем виде n-е число можно определить так:

$$F(n) = \begin{cases} 1, & \textit{если} \quad k = 1,\ n = 2, \\ F(n-1) + F(n-2), & \textit{если} \quad n > 2. \end{cases}$$

3. Для заданного натурального числа $N \geq 1$ определить единственное натуральное число a, для которого выполняется неравенство: $2^{a-1} \leq N \leq 2^a$.

4. Вычислить определитель матрицы, пользуясь формулой разложения по первой строке:

$$\det A = \sum_{1} (-1)^{1+1} a_{11} \cdot \det B_1,$$

где матрица B_k получается из A вычеркиванием первой строки и k-го столбца.

5. Напишите рекурсивную функцию для нахождения биномиальных коэффициентов (для заданного $M \geq i \geq j > 0$ вычислите все C_i^j):

$$C_m^n = \begin{cases} 1, & \textit{при } m = 0,\ n > 0 \textit{ или } m = n \geq 0, \\ 0, & \textit{при } m > n > 0, \\ C_{m-1}^{n-1} + C_m^{n-1}, & \textit{иначе.} \end{cases}$$

6. Пусть даны целое n от 2 до 20 и вещественное число $E > 0$. Найдите с точностью E все корни n-го многочлена Чебышева $T_n(x)$, определяемого формулами:
$T_0(x) = 1, \quad T_1(x) = x, \quad T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x), \quad (k = 1,2,....)$. *Примечание*: Многочлен $T_k(x)$ имеет k различных корней в интервале [-1,1]; если $x_1 < x_2 < ... < x_k$ - корни многочлена $T_k(x)$, то многочлен $T_{k+1}(x)$ имеет по одному корню в каждом из интервалов $[-1, x_1]$, $[x_1, x_2]$, …, $[x_k, 1]$.
7. Написать процедуру перевода числа из десятичной системы счисления в двоичную.
8. Напишите программу вычисления N-го члена последовательности, начинающейся с единицы в которой каждый следующий член равен сумме квадратов всех предыдущих.
9. Вычислить, используя рекурсию, выражение

$$\sqrt{6 + 2\sqrt{7 + 3\sqrt{8 + 4\sqrt{9 + \ldots}}}}$$

10. Для заданного целого n вычислите значение суммы:

$$\sum_{i_1=1}^{n} \sum_{i_2=1}^{n} \ldots \sum_{i_n=1}^{n} \frac{1}{i_1 + i_2 + \ldots + i_n}.$$

11. Опишите рекурсивную функцию, которая по заданным вещественному x и целому n вычисляет величину x^n согласно формуле:

$$x^n = \begin{cases} 1, & n = 0, \\ \dfrac{1}{x^{|n|}}, & n < 0, \\ x(x^{n-1}), & n > 0. \end{cases}$$

12. Напишите рекурсивную функцию, которая вычисляет $y = \sqrt[k]{x}$ по следующей формуле:

$y_0 = 1,\ y_{n+1} = y_n + \dfrac{\left(\dfrac{x}{y_n^{k-1}} - y_n\right)}{k},\ n = 0,1,2,...$ За ответ принять приближение, для которого выполняется условие $|y_n - y_{n+1}| < e,$ где $e = 0{,}0001.$

13. Для заданных границ интегрирования a и b вычислите значение определенного интеграла следующего вида:

$$\int \sin^n x\, dx = \begin{cases} -\dfrac{\sin^{n-1} x \cos x}{n} + \dfrac{n-1}{n}\int \sin^{n-2} x dx, & n > 2, \\ \dfrac{x}{2} - \dfrac{1}{4}\sin 2x, & n = 2, \\ -\cos x, & n = 1; \end{cases}$$

14. Для заданных границ интегрирования a и b вычислите значение определенного интеграла следующего вида:

$$\int \cos^n x\, dx = \begin{cases} \dfrac{\cos^{n-1} x \sin x}{n} + \dfrac{n-1}{n}\int \cos^{n-2} x dx, & n > 2, \\ \dfrac{x}{2} + \dfrac{1}{4}\sin 2x, & n = 2, \\ \sin x, & n = 1; \end{cases}$$

15. Для заданных границ интегрирования a и b вычислите значение определенного интеграла следующего вида:

$$\int \frac{dx}{\sin^n x} = \begin{cases} -\dfrac{1}{n-1} \cdot \dfrac{\cos x}{\sin^{n-1} x} + \dfrac{n-2}{n-1}\int \dfrac{dx}{\sin^{n-2} x}, & n \geq 2, \\ \ln tg \dfrac{x}{2}, & n = 1, \\ x, & n = 0; \end{cases}$$

16. Для заданных границ интегрирования a и b вычислите значение определенного интеграла следующего вида:

$$\int \frac{dx}{\cos^n x} = \begin{cases} \frac{1}{n-1} \cdot \frac{\sin x}{\cos^{n-1} x} + \frac{n-2}{n-1} \int \frac{dx}{\cos^{n-2} x}, & n \geq 2, \\ \ln tg\left(\frac{\pi}{4} + \frac{x}{2}\right), & n = 1, \\ x, & n = 0; \end{cases}$$

17. Для заданных границ интегрирования a и b вычислите значение определенного интеграла следующего вида:

$$\int x^n e^{ax} dx = \begin{cases} \frac{x^n e^{ax}}{a} - \frac{n}{a} \int x^{n-1} e^{ax} dx, & n > 1, \\ \frac{e^{ax}}{a^2}(ax - 1), & n = 1; \end{cases}$$

18. Для заданных границ интегрирования a и b вычислите значение определенного интеграла следующего вида:

$$\int x^n a^{mx} dx = \begin{cases} \frac{x^n a^{mx}}{n \ln a} - \frac{n}{m \ln a} \int x^{n-1} a^{mx} dx, & n > 1, \\ \frac{x a^{mx}}{m \ln a} - \frac{a^{mx}}{m (\ln a)^2}, & n = 1; \end{cases}$$

19. Получите все размещения из 10 элементов 1,2,…,10 по 3 в каждом. Размещением называется выборка из n указанных элементов m неповторяющихся элементов.
20. Определить максимальный элемент в массиве, используя рекурсивную процедуру для поиска максимума.
21. Вычислить величину Y=(2*n+1)!!*(2*m+1!!)/(2*(m+n)+1)!! , где m и n не отрицательные целые числа. Для определения (2*k+1)!! Использовать рекурсивную функцию.
22. Найти максимальную цифру в записи данного натурального числа.
23. Использовать рекурсию для нахождения цифрового корня целого числа. Цифровой корень находится суммой через сумму цифр числа до тех пор, пока эта сумма сама не станет цифрой. Например, для числа 9999999 цифровой корень находится так: 9+9+9+9+9+9+9 = 63; 6+3 = 9. Цифровой корень 9999999 равен девяти.
24. Напишите рекурсивную функцию, которая возвращает среднее из n элементов массива чисел.
25. Найти НОД (наибольший общий делитель) двух натуральных чисел.
26. Для данного n напечатайте коэффициенты разложения полинома (1 + x)^n.
27. Напишите рекурсивную функцию, которая вычисляет длину строки.
28. Проверить, является ли фрагмент строки с i-го по j-й символ палиндромом.

Лабораторная работа выполняется в четыре этапа.

1. *Подготовительный этап*, на котором описывается математическая модель алгоритма
2. *Основной этап,* на котором составляется алгоритм и программа работы, с использованием рекурсивной(ых) функции(й).
3. *Лабораторный этап*, на котором производится отладка программы.
4. *Составление отчета*. Отчет должен содержать:

- название лабораторной работы и текст варианта задания;
- листинг программы.

Для сдачи отчета необходимо продемонстрировать работу программы. Дополнительное требование к программе: корректность исходных данных, вводимых с клавиатуры, должна контролироваться

программой.

КОНТРОЛЬНЫЕ ВОПРОСЫ

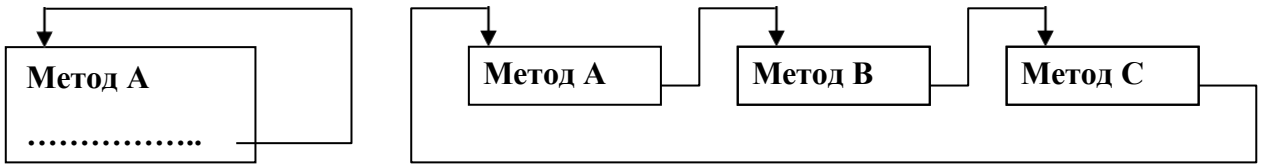
1. Из каких частей состоит рекурсивное определение?
2. Из каких частей состоит рекурсивный алгоритм?
3. Что такое прямая и косвенная рекурсия?
4. Сравните итеративную и рекурсивную организацию вычислительного процесса. Чем они отличаются?
5. Что такое бесконечная рекурсия? Какова причина ее возникновения?
6. Что такое уровень рекурсии? Что такое глубина рекурсии?
7. Каким образом используются фреймы активации в процессе работы рекурсивного алгоритма?
8. Что такое бэктрекинг? Как реализуются алгоритмы поиска с возвратом?

Методические рекомендации

Рекурсивное определение алгоритма включает две части:

1. ***условия завершения*** (одно или несколько), которые могут быть вычислены для определенных параметров. Условия завершения соответствуют базисной части рекурсивного определения;
2. ***шаг рекурсии***, в котором текущие значения некоторых переменных в алгоритме могут быть определены с использованием их предыдущих значений. В конечном итоге шаг рекурсии должен приводить к выполнению условий завершения.

Рекурсивные алгоритмы реализуются через ***рекурсивные методы (функции)***. Рекурсивным называется метод, который в процессе выполнения явно или неявно вызывает сам себя. ***Прямая (явная) рекурсия*** характеризуется наличием в теле метода оператора обращения к нему же самому (рис. 1.а). В случае ***косвенной (неявной) рекурсии*** один метод обращается к другому, который (возможно через цепочку вызовов других методов) вновь обращается к первому методу (рис. 1.б). Далее будем рассматривать только прямую рекурсию.



а **б**

Рис. 1. Схема прямой и косвенной рекурсии: а – прямая рекурсия; б – косвенная рекурсия

Итеративная и рекурсивная схема организации вычислительного процесса

Для того чтобы лучше понять особенности рекурсивных алгоритмов, полезно сопоставить итеративную и рекурсивную организацию процесса вычислений в программе. Особенности итеративного и рекурсивного вычислительного процесса рассмотрим на примере вычисления значения факториала некоторого натурального числа *N*.

Итеративная схема организации вычислительного процесса

Итеративный процесс можно проиллюстрировать с помощью схемы, приведенной на рис. 2. Этот процесс состоит из четырех блоков: инициализации, принятия решения (о продолжении вычислений), вычисления и модификации.

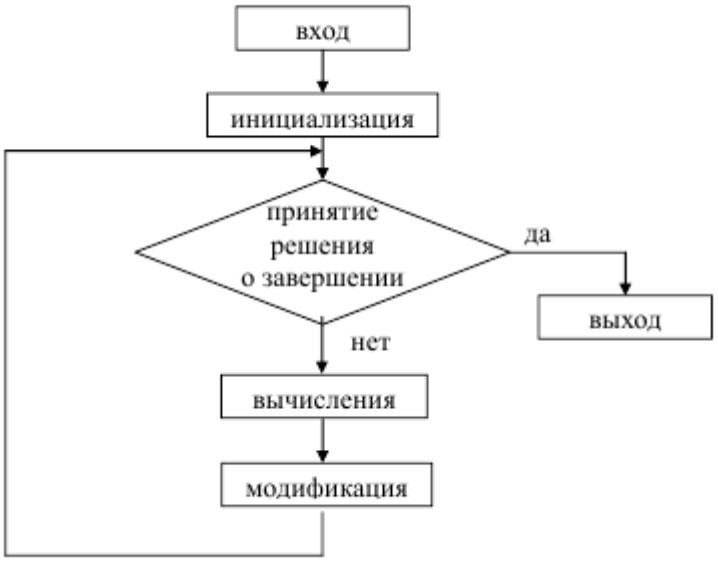


Рис. 2. Схема итеративного вычислительного процесса

В основе итеративного вычислительного процесса лежит ***итеративный цикл*** *While (с предусловием или постусловием), For*.
Наиболее универсальным является цикл *While*:
While < условие цикла > < тело цикла >;

Итеративная схема вычисления факториала:
N! = 1 * 2 * 3 * … * N.
Метод, реализующий итеративную схему вычисления факториала, приведен ниже:

```
using System;
namespace Iteration
{
class Iteration
{
public static long Iter_fact( int n )
{
int i =1; long f =1;                                    // инициализация
while ( i < = n )                                       // решение о завершении
{
f = f * i;                                              // вычисления
i++;                                                    // модификация
    }
    return f;
    }

    static void Main()
    {
    Console.Write( “Введите целое число: “ );
    int x = Convert.ToInt32(( Console.ReadLine() );
    long y = Iter_fact( x);
```

```
  Console.WriteLine( “Итеративный факториал = {0}”, y );
  }
  }
}
```

Существует два важных положения, известных в математике и в программировании, определяющих соотношение между итерацией и рекурсией:

1. любой итеративный цикл может быть заменен рекурсией;
2. рекурсия не всегда может быть заменена итерацией.

Рекурсивная схема организации вычислительного процесса

Общая схема рекурсивного вычислительного процесса представлен

Рис. 3. Схема рекурсивного вычислительного процесса

Так как обращаться к рекурсивному методу можно как из него самого, так и извне, каждое обращение к рекурсивному методу вызывает его независимую активацию. При каждой активации образуются копии всех локальных переменных и формальных параметров рекурсивного метода, в которых “оставляют следы” операторы текущей активации. Таким образом, для рекурсивного метода может одновременно существовать несколько активаций. Для обеспечения правильного функционирования рекурсивного метода необходимо сохранять адреса возврата в таком порядке, чтобы возврат после завершения каждой текущей активации выполнялся в точку, соответствующую оператору, непосредственно следующему за оператором рекурсивного вызова. Совокупность локальных переменных, значений фактических параметров, подставляемых на место формальных параметров рекурсивного метода, и адреса возврата однозначно характеризует текущую активацию и образует ***фрейм активации***. Фрейм активации необходимо сохранять при очередной активации и восстанавливать после завершения текущей активации.
В блоке принятия решения (о продолжении вычислений) производится проверка, являются ли значения входных параметров такими, для которых возможно вычисление значений выходных параметров в соответствии с базисной частью рекурсивного определения. На основании этой проверки принимается решение о выполнении промежуточных или окончательных вычислений. Блок промежуточных вычислений можно объединить с блоком обращения к методу, если промежуточные вычисления очень просты. В блоке окончательных вычислений производится явное определение параметров-переменных метода для конкретных значений входных параметров, соответствующих текущей активации метода.
В основе рекурсивного вычислительного процесса лежит ***рекурсивный цикл***, который реализуется через вызов рекурсивного метода, причем каждая активация рекурсивного метода эквивалентна одному проходу итеративного цикла *While*.
Общая схема рекурсивного цикла:

```
[спецификаторы] тип Recursion_Cycle (...)
{
```

```
if < условие цикла >
{
< тело рекурсивного цикла; > Recursion Cycle (…);
}
}
```

В теле рекурсивного цикла (в блоке промежуточных вычислений) обязательно должны содержаться операторы, изменяющие значения переменных, от которых зависит условие завершения рекурсивного цикла.

Выполнение условия завершения рекурсивного цикла соответствует достижению базиса рекурсивного определения. Если значения этих переменных не успевают измениться в теле цикла до очередной активации рекурсивного метода, то возникает ***бесконечный рекурсивный цикл***.

Общая схема бесконечного рекурсивного цикла:

```
[спецификаторы] тип Infinite Recursion_Cycle (...)
{
if < условие цикла >
{
Infinite Recursion_Cycle (…);
< тело рекурсивного цикла; >
}
}
```

Рекурсивная схема вычисления факториала:

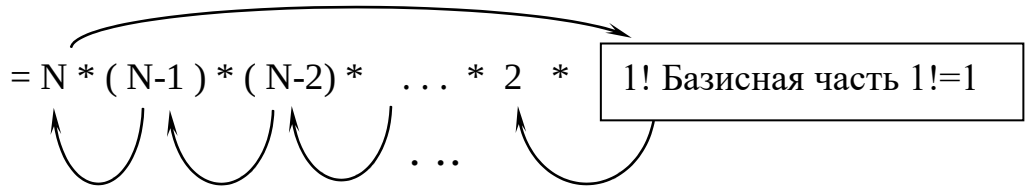
Базисная часть:

0! = 1; 1! = 1;

Рекурсивная часть:

N! = N * (N-1)! = N * (N-1) * (N-2)! = N * (N-1) * … * (N – (N-1))! =

Промежуточные вычисления и обращения метода к самому себе



Окончательные вычисления

Метод, реализующий рекурсивную схему вычисления факториала:

```
using System;
namespace Recursion
{
class Recursion
{
public static long Recurs_fact( int n )
{
long f;
if ( n == 0 || n == 1)                    // принятие решения о завершении вычислений:
f = 1;                                    // да – окончательные вычисления для базисной части
    else
```

```
{
f = Recurs_fact ( n-1);                                  // нет – промежуточные вычисления и
                                                          обращение метода к самому себе
f = f * n;                    // окончательные вычисления для рекурсивной части { адрес
                              возврата после завершения активации }
            }
return f;
}                                                        // завершение активации

static void Main()
{
Console.Write( “Введите целое число: “ );
int x = int.Parse( Console.ReadLine() );
long y = Recurs_fact( x );
Console.WriteLine( “Рекурсивный факториал = {0}”, y );
}
}
}
```

С каждым обращением к рекурсивному методу ассоциируется ***номер уровня рекурсии*** (номер фрейма активации). Считается, что при первоначальном вызове рекурсивного метода из основной программы номер уровня рекурсии равен единице. Каждый следующий вход в метод имеет номер уровня на единицу больше, чем номер уровня метода, из которого производится это обращение.

Другой характеристикой рекурсивного метода является ***глубина рекурсии***, определяемая максимальным уровнем рекурсии в процессе вычисления при заданных аргументах. В общем случае эта величина неочевидна, исключение составляют простые рекурсивные методы, например, при вычислении значения N! глубина рекурсии равна N.

Так как при выходе из текущей активации самым первым должен быть восстановлен фрейм, который был позже всех сохранен, для хранения фреймов используется область системного стека. Рисунок 4 поясняет механизм рекурсивных вычислений.

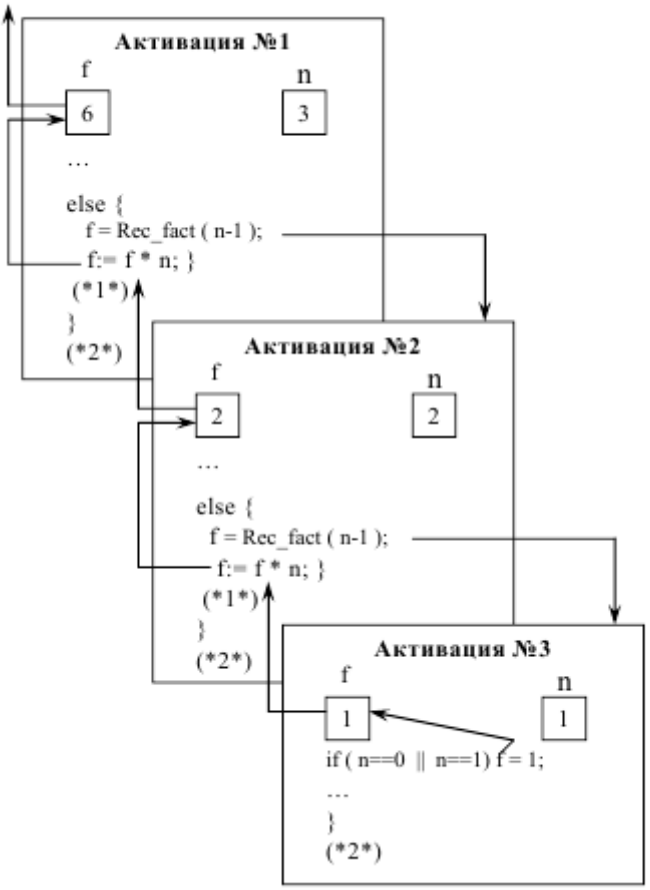


Рис. 4. Фреймы активации при вычислении 3!