Лабораторная работа №1. Рекурсивные функции

Цель работы: изучить способы реализации алгоритмов с использованием рекурсии.

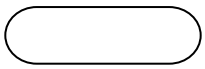
1.1. Краткие теоретические сведения

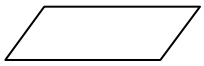
Рекурсия – это способ организации вычислительного процесса, при котором функция в ходе выполнения, входящих в нее операторов обращается сама к себе. Классическим примером является вычисление факториала $n!$ ($n > 0$):

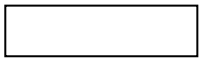
```
double Faktorial_R (int n)  {
 if (n < 2) return 1;                    // Условие окончания рекурсии
 else return n* Faktorial_R (n–1);       // Рекурсивное обращение к функции
}
```

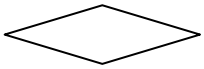
При выполнении правильно организованной рекурсивной функции осуществляется последовательный переход от текущего уровня организации алгоритма к нижнему уровню, в котором будет получено *нерекурсивное* решение задачи (в приведенном примере при $n < 2$), т.е. не требующее дальнейшего обращения к функции.

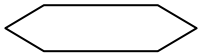
При описании алгоритмов используем следующие стандартные фигуры блок-схем:

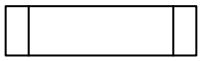

– начало функции (заголовок) и конец функции (выход и возвращение в точку вызова);


– ввод-вывод данных;


– вычислительный процесс (операция присваивания);


– блок сравнения (операторы *if, switch-case*);


– заголовок оператора цикла;


– обращение к функции пользователя.

1.2. Пример выполнения задания

Написать программу вычисления факториала ***положительного*** числа ***n***, содержащую функции пользователя с рекурсией и без рекурсии.

1.2.1. Реализация задания в оконном приложении

Вид формы и полученные результаты представлены на рис. 1.1. Компонента *Edit*1 используется для ввода *n*, а компоненты *Edit*2 и *Edit*3 – для вывода результатов.

Листинг программы может иметь следующий вид:

Блок-схема функции-обработчика *Button1Click* представлена на рис. 1.2.

```
                . . .
            double Faktorial(int);
            double Faktorial_R(int);
//-------------------- Кнопка START ----------------------------------------
void __fastcall TForm1::Button1Click(TObject *Sender)
{
        int n = StrToInt(Edit1->Text);
        switch(RadioGroup1->ItemIndex) {
            case 0:
                Edit2->Text = FloatToStrF(Faktorial_R(n), ffFixed, 8, 1);
            break;
            case 1:
                Edit3->Text = FloatToStrF(Faktorial(n), ffFixed, 8, 1);
            break;
        }
}
//------------------ Функция без рекурсии -------------------------------------
double Faktorial(int n)
{
            double f = 1;
                for (int i = 1; i <= n; i++) f  *=  i;
                return f;
}
//------------------- Рекурсивная функция ---------------------------------------
double Faktorial_R(int n)
{
        if (n < 2) return 1;
        else
                return n*Faktorial_R(n-1);
}
```

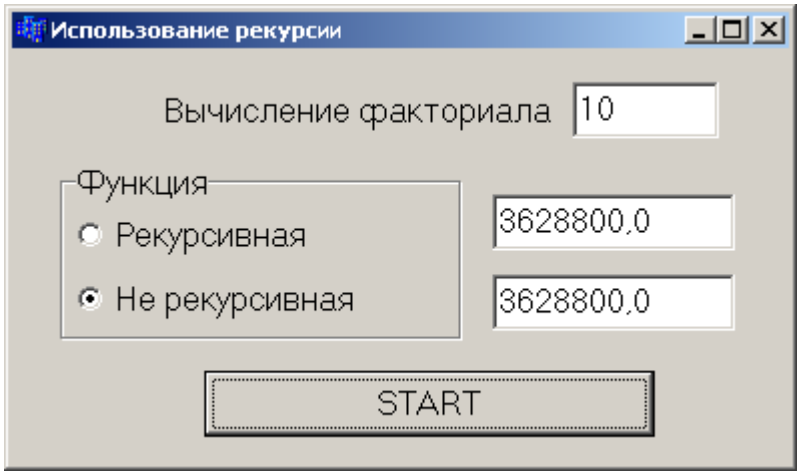


Рис. 1.1

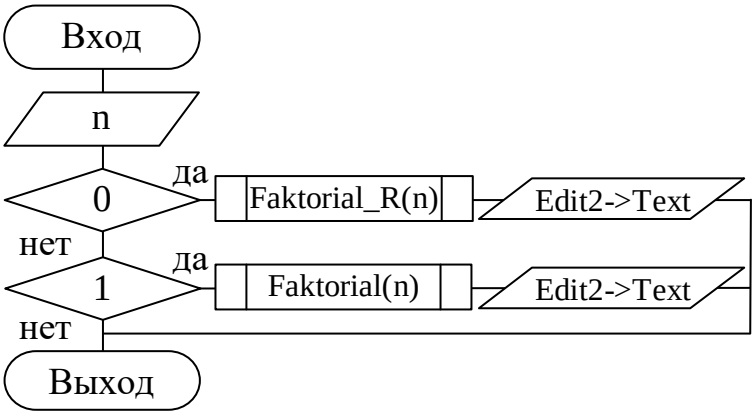


Рис. 1.2

Блок-схемы функций пользователя *Faktorial_R* и *Faktorial* представлены на рис. 1.3.

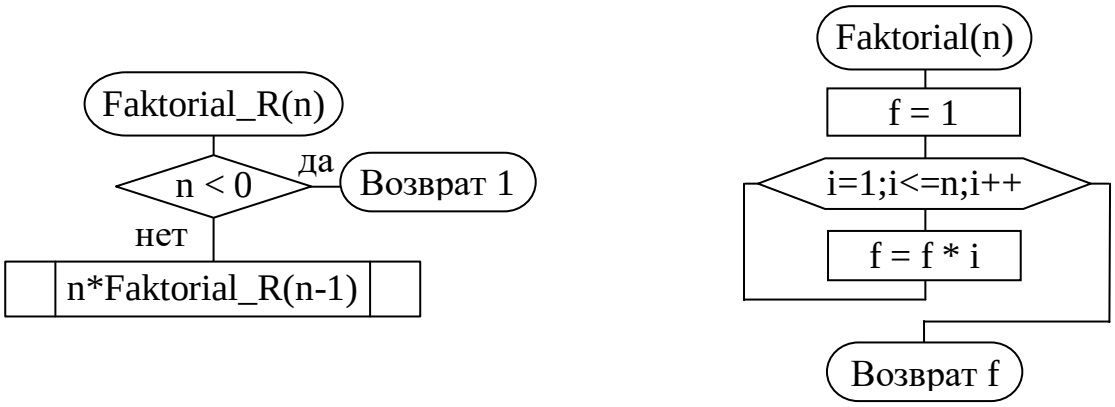


Рис. 1.3

1.2.2. Реализация задания в консольном приложении

Тексты функций пользователя смотрите в предыдущем примере, а листинг основной функции может иметь следующий вид:

```
        …
#include <iostream.h>
#include <iomanip.h>                    // Для использования setprecision(n)
        …
double Faktorial(int);
double Faktorial_R(int);
void  main(void)
{
        int n, kod;
        while(true) {                   // Бесконечный цикл
                cout << "\n Recurs - 0\n Simple - 1\n Else - Exit\t";
                cin >> kod;
                if (kod <0 || kod > 1) return;
                cout << "\tInput n " ;
                cin >> n;
                switch(kod) {
                        case 0:
```

```
            cout << setprecision(10) << "\tRecurs = " << Faktorial_R(n) << endl;
        break;
        case 1:
            cout << setprecision(10) << "\tSimple = " << Faktorial(n) << endl;
        break;
        }
    }
}
```

Результаты выполнения программы представлены на рис. 1.4:

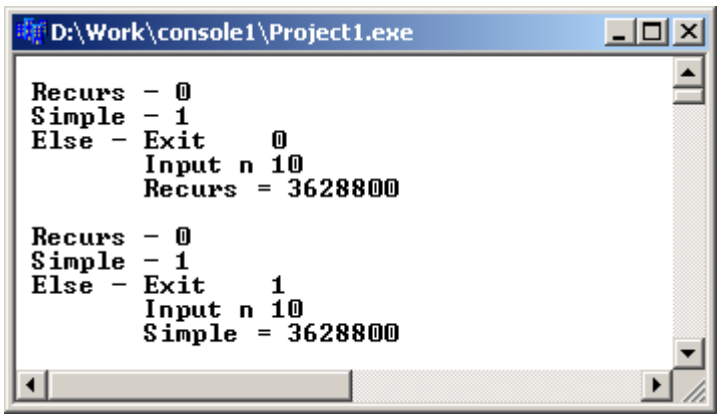


Рис. 1.4

1.3. Индивидуальные задания

Составить алгоритм в виде блок-схемы, написать и отладить поставленную задачу с использованием рекурсивной и обычной функций. Сравнить полученные результаты.

1. Для заданного целого десятичного числа N получить его представление в p-ичной системе счисления (p <10).

2. В упорядоченном массиве целых чисел a_i (i = 1, ..., n) найти номер находящегося в массиве элемента c, используя метод двоичного поиска.

3. Найти наибольший общий делитель чисел M и N, используя теорему Эйлера: если M делится на N, то $НОД\ (N,M) = N$, иначе $НОД\ (N,M) = (M\%N, N)$.

4. Числа Фибоначчи определяются следующим образом: $Fb(0) = 0$; $Fb(1) = 1$; $Fb(n) = Fb(n–1) + Fb(n–2)$. Определить $Fb(n)$.

5. Найти значение функции Аккермана $A(m, n)$, которая определяется для всех неотрицательных целых аргументов m и n следующим образом:

$A(0, n) = n + 1$;

$A(m, 0) = A(m–1, 1)$; при $m > 0$;

$A(m, n) = A(m–1, A(m, n–1))$; при $m > 0$ и $n > 0$.

6. Найти методом деления отрезка пополам минимум функции $f(x) = 7\sin^2(x)$ на отрезке [2, 6] с заданной точностью ε (например 0,01).

7. Вычислить значение $x = \sqrt{a}$, используя рекуррентную формулу $x_n = \frac{1}{2}\left(x_{n-1} + \frac{a}{x_{n-1}}\right)$, в качестве начального значения использовать $x_0 = 0{,}5\cdot(1 + a)$.

8. Найти максимальный элемент в массиве a_i (i=1, …, n), используя очевидное соотношение $max(a_1, \ldots, a_n) = max[max(a_1, \ldots, a_{n-1}), a_n]$.

9. Вычислить значение $y(n) = \sqrt{1+\sqrt{2+\ldots+\sqrt{n}}}$.

10. Найти максимальный элемент в массиве a_i (i=1, …, n), используя соотношение (деления пополам) $max(a_1,\ldots, a_n) = max[max(a_1,\ldots, a_{n/2}), max(a_{n/2+1}, \ldots, a_n)]$.

11. Вычислить значение $y(n) = \cfrac{1}{n+\cfrac{1}{(n-1)+\cfrac{1}{(n-2)+\cfrac{1}{\cfrac{\ldots}{\ldots+\cfrac{1}{1+\cfrac{1}{2}}}}}}}$.

12. Вычислить произведение четного количества n ($n \geq 2$) сомножителей следующего вида:

$$y = \left(\frac{2}{1}\cdot\frac{2}{3}\right)\cdot\left(\frac{4}{3}\cdot\frac{4}{5}\right)\cdot\left(\frac{6}{5}\cdot\frac{6}{7}\right)\cdot\ldots .$$

13. Вычислить $y = x^n$ по следующему правилу: $y = (x^{n/2})^2$, если n четное и $y = x \cdot y^{n-1}$, если n нечетное.

14. Вычислить значение $C_n^k = \frac{n!}{k!(n-k)!}$ (значение 0! = 1).

15. Вычислить $y(n) = \cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{\ldots}}}}$, n задает число ступеней.

16. В заданном массиве заменить все числа, граничащие с цифрой «1», нулями.