//Вывод числа N слева на право

#include <iostream>

using namespace std;

void rekursion01(int n, int i = 1)

{

if (n == 0) return;

else

{

cout << (n - n) + i<< " ";

return rekursion01(n - 1, i+1);

}

}

void main()

{

rekursion01(10);

system("pause");

}

//Factorial

#include <iostream>

using namespace std;

int rekursion01(int n)

{

if (n == 1) return 1;

else return n\*rekursion01(n-1);

}

void main()

{

cout << rekursion01(6) << endl;

system("pause");

}

//Возведение в степени

#include <iostream>

using namespace std;

int rekursion01(int n, int m)

{

if (m == 1) return 1;

else return n\*rekursion01(n, m-1);

}

void main()

{

cout << rekursion01(2,9) << endl;

system("pause");

}

//Сумма цифр числа

#include <iostream>

using namespace std;

int rekursion01(int n)

{

if (n == 0) return 0;

else

{

return n%10 + rekursion01(n/10);

}

}

void main()

{

cout << rekursion01(179) << endl;

system("pause");

}

//Количество цифр в числе

#include <iostream>

using namespace std;

int rekursion01(int n, int counter=0)

{

if (n == 0) return 0;

else

{

counter++;

return counter+rekursion01(n/10);

}

}

void main()

{

cout << rekursion01(2569) << endl;

system("pause");

}

//Digit root – Цифровой корень

#include <iostream>

using namespace std;

template <typename T>

T sum\_of\_digits(T n)

{

if (n == 0) return 0;

else

return n % 10 + sum\_of\_digits(n / 10);

}

template <typename K>

K digit\_root(K n)

{

K our\_sum\_of\_digits = sum\_of\_digits(n);

if (our\_sum\_of\_digits <= 9)

return our\_sum\_of\_digits;

else

return digit\_root(our\_sum\_of\_digits);

}

void main()

{

cout << digit\_root(22483) << endl;

system("pause");

}

//Арифметическая прогрессия – найти сумму заданного шага прогрессии

int AP(int a, int b, int c)

{

if (c == 1)

return a;

else

return AP(a+b, b, c-1);

}

void main()

{

cout << AP(3,2,8) << endl;

system("pause");

}

//Hailstone numbers

void Hailstone(int n)

{

cout << n << " ";

if (n == 1) return;

else

{

if (n % 2 == 0) return Hailstone(n / 2);

if (n % 2 != 0) return Hailstone(n \* 3 + 1);

}

}

void main()

{

Hailstone(7);

system("pause");

}

//From decimal to binary (только задом наперед вывод получается)

void from\_dec\_to\_bin(int n)

{

if (n == 0) return;

else

{

cout << n%2 << " ";

return from\_dec\_to\_bin(n / 2);

}

}

void main()

{

from\_dec\_to\_bin(32);

system("pause");

}