**Практична робота № 19-20**

**Завдання №1.** Введіть програму 1, запустіть її для n = 5 і поясніть отримані результати.

#include <iostream>

#include <conio.h>

using namespace std;

unsigned long int factorial(unsigned long int);

int i = 1;

unsigned long int result;

int main()

{

int n;

cout << "Enter n!: ";

cin >> n;

cout << n << "!" << "=" << factorial(n) << endl;

system("pause");

return 0;

}

unsigned long int factorial(unsigned long int f){

if(f == 1 || f == 0)

return 1;

cout << "Step\t" << i << endl;

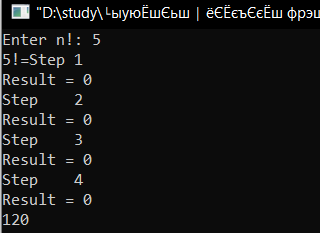
i++;

cout << "Result = " << result << endl;

result = f \* factorial(f - 1);

return result;

}



*Рекурсивне вирішення факторіалу – функція виклакає саму себе рекурсивно, допоки не знайде базове рішення*

**Завдання №2.** Введіть програму 2, запустіть її для n = 20 і поясніть отримані результати.

#include <iostream>

#include <conio.h>

using namespace std;

unsigned long int factorial(unsigned long int);

int i = 1;

unsigned long int result;

int main()

{

int n;

cout << "Enter n for calculating n!: ";

cin >> n;

for(int k = 1; k <= n; k++)

{

cout << k << "!" << "=" << factorial(k) << endl;

}

system("pause");

return 0;

}

unsigned long int factorial(unsigned long int f){

if(f == 1 || f == 0)

return 1;

//cout << "Step\t" << i << endl;

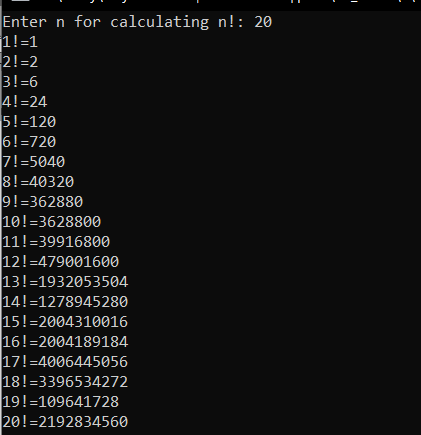
i++;

//cout << "Result = " << result << endl;

result = f \* factorial(f - 1);

return result;

}



*Максимальне значення для змінної типу unsigned long - 4 294 967 295. Через це вирахування факторіалу числа 20 не можливе.*

**Завдання №3.** Самостійно складіть програму обчислення факторіалу без застосування рекурсії і порівняйте її з результатами попередньої програми.

#include <iostream>

#include<conio.h>

using namespace std;

unsigned long int factorial(unsigned long int);

int i = 1;

unsigned long int result = 1;

int main()

{

int n;

cout << "Enter n!"<< endl;

cin >> n;

factorial(n);

system("pause");

return 0;

}

unsigned long int factorial(unsigned long int f) {

if (f == 1 || f == 0) {

return 1;

}

for (int i = 1; i != f; i++) {

result = result + result \* i;

}

cout << f << "!" << " = " << result << endl;;

}



**Завдання №4.** Введіть програму 3, обчисліть числа Фібоначчі до n = 30 і перевірте отримані результати.

#include <iostream>

#include <conio.h>

using namespace std;

unsigned long fibonacci(unsigned long);

int main()

{

unsigned long entered\_number;

cout << "Enter number from the Fibonacci series: ";

cin >> entered\_number;

for(int counter = 1; counter <= entered\_number; counter++)

cout <<counter << " = "<< fibonacci(counter) << endl;

system("pause");

return 0;

}

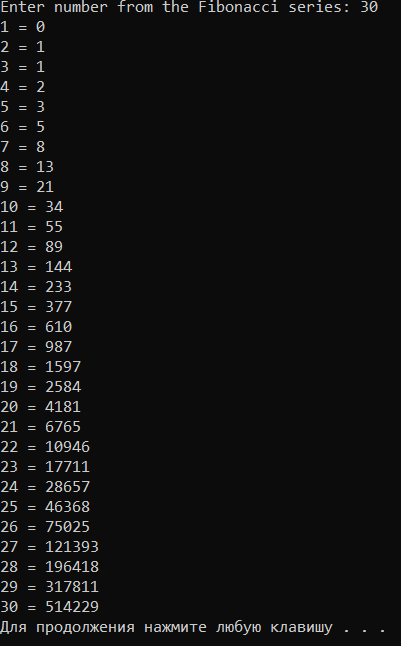
unsigned long fibonacci(unsigned long entered\_number)

{

if(entered\_number == 1 || entered\_number == 2) return (entered\_number - 1);

return fibonacci(entered\_number - 1) + fibonacci(entered\_number - 2);

}



**Завдання №5.** Самостійно складіть програму обчислення чисел Фібоначчі, перевірте її для n = 30 і порівняйте результати з попередньою програмою.

#include <iostream>

#include <conio.h>

using namespace std;

using namespace std;

int main()

{

int a, b, k, n = 0;

a = 0;

b = 1;

cout << "Enter number from Fibonacci series: ";

cin >> k;

while (k <= 0)

{

cout << "ERROR! Enter the number greater than zero: ";

cin >> k;

}

while (k != 1)

{

n++;

a = a + b;

b = a - b;

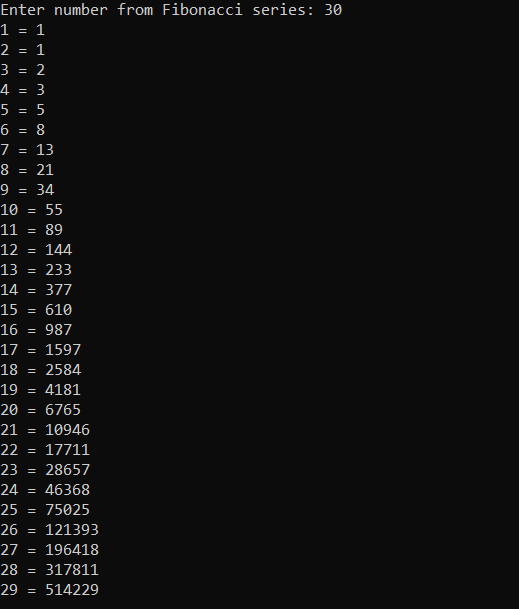
k = k - 1;

cout << n << " = "<< a << endl;

}

return 0;

}



**Завдання №6.** Вивчіть гру Ханойська вежа для n = 3 і n = 5, запустіть програму 4 і перевірте її роботу для тих самих значень кількості дисків.

#include <iostream>

#include <conio.h>

using namespace std;

void tower(int, int, int, int);

int count1 = 1;

int main()

{

cout << "Enter of number of disks: ";

int number;

cin >> number;

cout << "Enter the number of basic rod: ";

int basic\_rod;

cin >> basic\_rod;

cout << "Enter the number of final rod: ";

int final\_rod;

cin >> final\_rod;

int help\_rod;

if(basic\_rod != 2 && final\_rod != 2){

help\_rod = 2;

}

else{

if(basic\_rod != 1 && final\_rod != 1) help\_rod = 1;

else{

if(basic\_rod != 3 && final\_rod != 3) help\_rod = 3;

}

}

tower(number, basic\_rod, help\_rod, final\_rod);

getch();

return 0;

}

void tower(int count\_disk, int baza, int help\_baza, int new\_baza)

{

if(count\_disk == 1)

{

cout <<count1<<") "<< baza << " " << "->" << " " << new\_baza << endl;

count1++;

}

else

{

tower(count\_disk - 1, baza, new\_baza, help\_baza);

tower(1, baza, help\_baza, new\_baza);

tower(count\_disk - 1, help\_baza, baza, new\_baza);

}

}

