Task 01)

#include <iostream>

using namespace std;

int power(int x, int n) {

if ( n < 0) {

return 0;

}

return pow(x, n);

}

void print(string name, int x, int n, int result, bool flag) {

cout << name << (flag ? " \002 PASS" : " \001 FAIL")

<< ": with x = " << x << "and n = " << n << " result is " << result << endl;

}

void test01() {

int x = 2;

int n = -1;

int expected = 0;

int actual = power(x, n);

print("test01", x, n, actual, expected == actual);

}

void test02() {

int x = 2;

int n = 0;

int expected = 1;

int actual = power(x, n);

print("test02", x, n, actual, expected == actual);

}

void test03() {

int x = 2;

int n = 3;

int expected = 8;

int actual = power(x, n);

print("test03", x, n, actual, expected == actual);

}

void test04() {

int x = 3;

int n = 2;

int expected = 9;

int actual = power(x, n);

print("test04", x, n, actual, expected == actual);

}

void test05() {

int x = -2;

int n = 3;

int expected = -8;

int actual = power(x, n);

print("test05", x, n, actual, expected == actual);

}

void test06() {

int x = -2;

int n = 2;

int expected = 4;

int actual = power(x, n);

print("test06", x, n, actual, expected == actual);

}

void test07() {

int x = -2;

int n = 0;

int expected = 1;

int actual = power(x, n);

print("test07", x, n, actual, expected == actual);

}

void test08() {

int x = 0;

int n = 0;

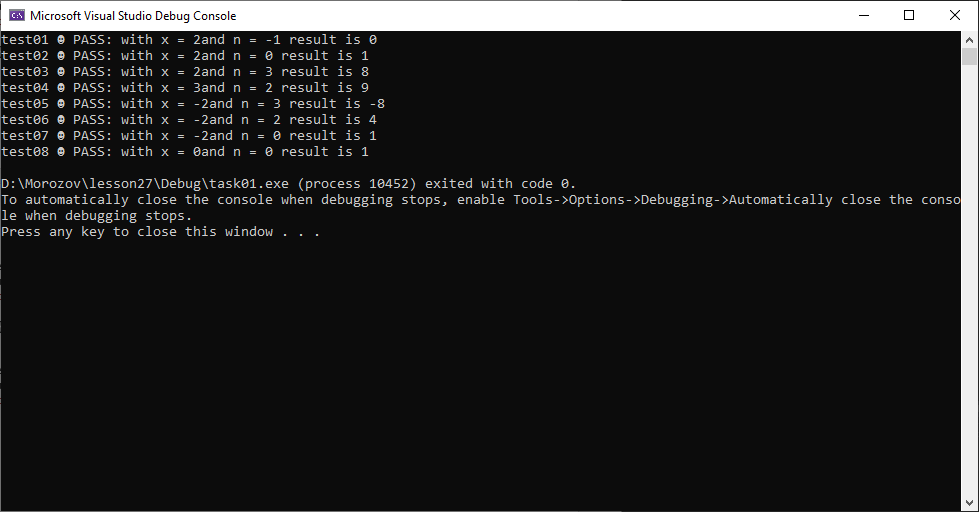
int expected = 1;

int actual = power(x, n);

print("test08", x, n, actual, expected == actual);

}

Result:



Task 02)

Result

Task 03)

#include <iostream>

using namespace std;

bool power\_of\_two(int number) {

for (int i = 1; pow(2, i) <= number; i++) {

if (pow(2, i) == number) {

return true;

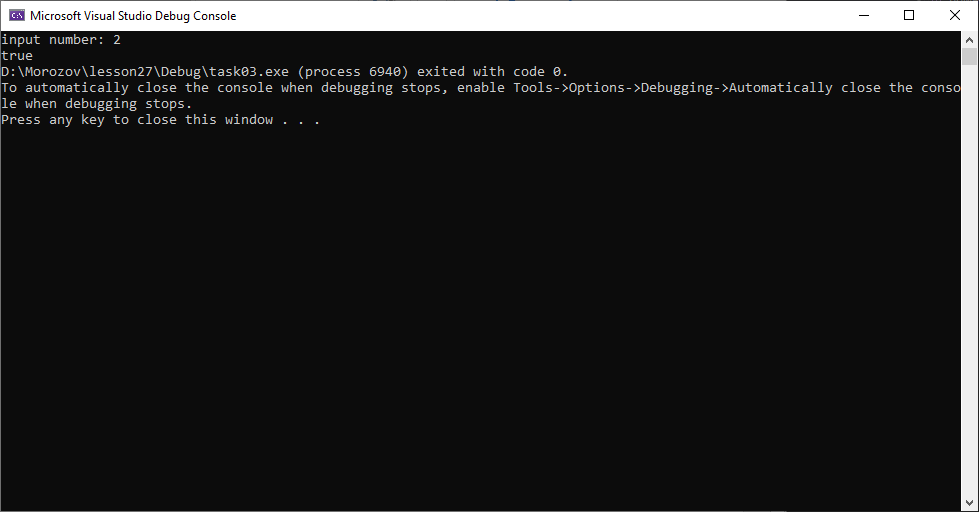
}

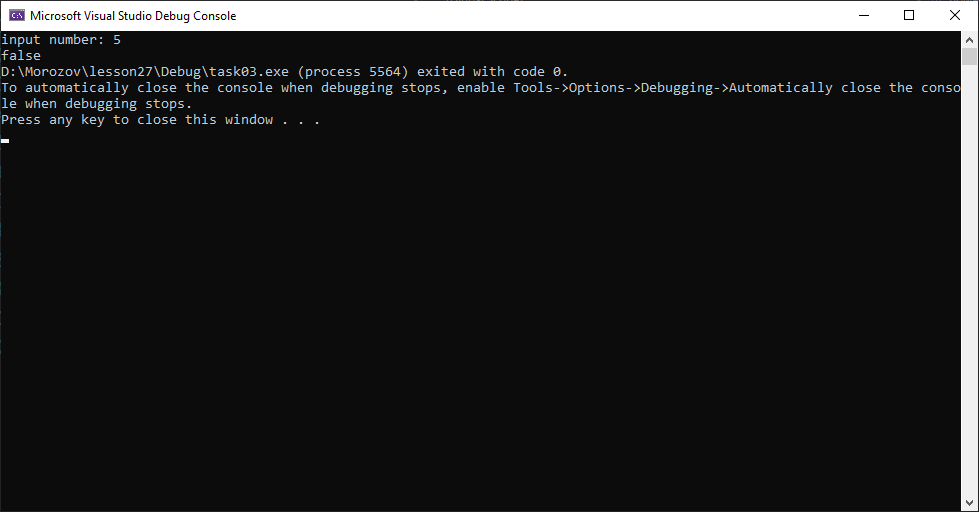
}

return false;

}

Result





Task 04)

#include <iostream>

using namespace std;

bool is\_digits\_count\_even(int number) {

if (number < 0) {

number \*= -1;

}

int num;

for ( num = 0; number != 0; number /= 10) {

if (number % 10 > 0) {

num++;

}

}

if (num % 2 == 0) {

return true;

}

return false;

}

void print(string name, long long number, int result, bool flag) {

cout << name << (flag ? " \002 PASS" : " \001 FAIL")

<< ": with number = " << number << " result is "

<< (result == 1 ? "true" : "false") << endl;

}

void test01() {

int number = 2;

bool expected = false;

bool actual = is\_digits\_count\_even(number);

print("test01", number, actual, expected == actual);

}

void test02() {

int number = -2;

bool expected = false;

bool actual = is\_digits\_count\_even(number);

print("test02", number, actual, expected == actual);

}

void test03() {

int number = 22;

bool expected = true;

bool actual = is\_digits\_count\_even(number);

print("test03", number, actual, expected == actual);

}

void test04() {

int number = -22;

bool expected = true;

bool actual = is\_digits\_count\_even(number);

print("test04", number, actual, expected == actual);

}

void test05() {

int number = 0;

bool expected = false;

bool actual = is\_digits\_count\_even(number);

print("test05", number, actual, expected == actual);

}

void test06() {

int number = 1234567;

bool expected = false;

bool actual = is\_digits\_count\_even(number);

print("test06", number, actual, expected == actual);

}

void test07() {

int number = 234567;

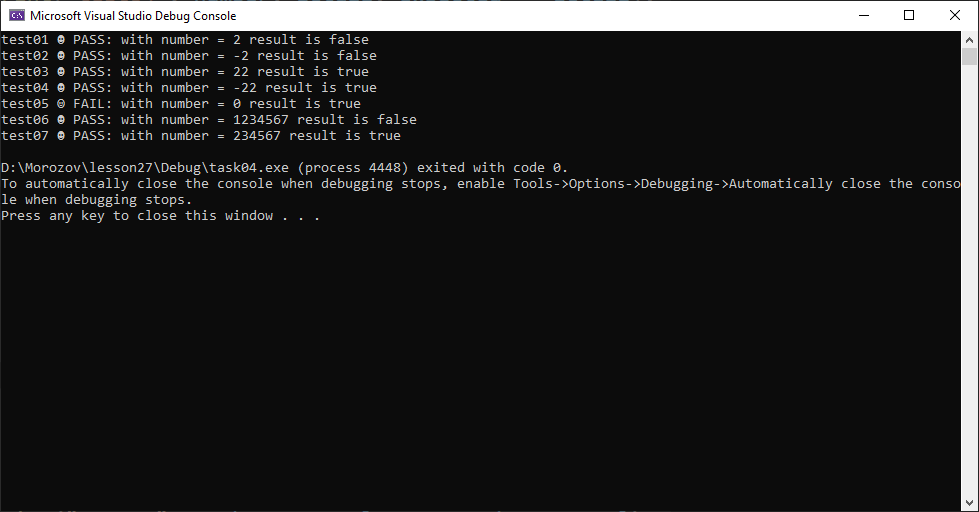
bool expected = true;

bool actual = is\_digits\_count\_even(number);

print("test07", number, actual, expected == actual);

}

Result



Task 05)

#include <iostream>

using namespace std;

int get\_fibo\_number(int number) {

if (number <= 3) {

return number == 1 ? 0 : 1;

}

int sum;

int fibo1 = 0;

int fibo2 = 1;

int fibo3 = fibo1 + fibo2;

int fibo = fibo2 + fibo3;

for (int i = 0;

i <= number; i++) {

sum = fibo + fibo3;

if (i + 1 < number) {

swap(sum, fibo);

swap(fibo, fibo3);

}

}

return sum;

}

Task06)

#include <iostream>

#include <string>

using namespace std;

bool prime\_number(int num) {

for (int i = 2; i < num; i++) {

if (num % i == 0) {

return false;

}

}

return true;

}

void get\_prime\_numbers(int number) {

for (int i = 2; i <= number; i++) {

cout << (prime\_number(i) == true ? to\_string(i)+" " : "");

}

}

Result

