Выполнил студент группы 348001:  
Паращенко Никита Васильевич

**Условие**  
Реализовать алгоритм вычисления Наибольшоего Общего Делителя двух чисел нативным и «правильным» алгоритмом. Замерить время выполнения каждого из алгоритмов для следующих чисел:

* 8723462536745223 и 765412221883
* 2973948729384212 и 87268364827634
* 1098304287349821 и 125367765723
* 2987349872934234 и 78238486224

**Решение (на Delphi)**

**Нативный алгоритм:**

program GCDNative;

uses

System.SysUtils, System.Diagnostics;

function GCDRecursive(A, B: Int64): Int64;

begin

if B = 0 then

Result := A

else

Result := GCDRecursive(B, A mod B);

end;

procedure time;

const

TestValues: array[0..3, 0..1] of Int64 = (

(8723462536745223, 765412221883),

(2973948729384212, 87268364827634),

(1098304287349821, 125367765723),

(2987349872934234, 78238486224)

);

RepeatCount = 10000; // Количество повторов для точности измерений

var

I, J: Integer;

Stopwatch: TStopwatch;

TotalTime, AvgTime: Double;

begin

Writeln('A', #9, 'B', #9, 'Recursive (ms)');

for I := 0 to High(TestValues) do

begin

Writeln('Testing A = ', TestValues[I, 0], ', B = ', TestValues[I, 1]);

TotalTime := 0;

for J := 1 to RepeatCount do

begin

Stopwatch := TStopwatch.StartNew;

GCDRecursive(TestValues[I, 0], TestValues[I, 1]);

Stopwatch.Stop;

TotalTime := TotalTime + Stopwatch.ElapsedTicks;

end;

AvgTime := (TotalTime / RepeatCount) \* 1000 / Stopwatch.Frequency; // Время в миллисекундах

Writeln(TestValues[I, 0], #9, TestValues[I, 1], #9, FormatFloat('0.000', AvgTime));

end;

end;

begin

try

time;

except

on E: Exception do

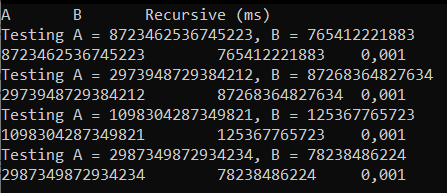
Writeln('Error: ', E.Message);

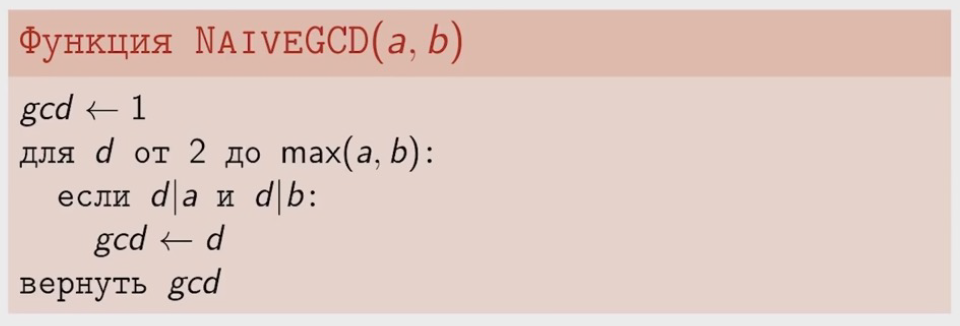
end;

Readln;

end.

**Время выполнения:**





**«Правильный алгоритм»:**

program GCDSeffective;

uses

System.SysUtils, System.Diagnostics;

function EuclidGCD(a, b: Int64): Int64;

begin

if a = 0 then

Result := b

else if b = 0 then

Result := a

else if a >= b then

Result := EuclidGCD(a mod b, b)

else

Result := EuclidGCD(a, b mod a);

end;

procedure time;

const

TestValues: array[0..3, 0..1] of Int64 = (

(8723462536745223, 765412221883),

(2973948729384212, 87268364827634),

(1098304287349821, 125367765723),

(2987349872934234, 78238486224)

);

RepeatCount = 10000; // Количество повторений для точности

var

I, J: Integer;

Stopwatch: TStopwatch;

TotalTime, AvgTime: Double;

begin

Writeln('A', #9, 'B', #9, 'Effective GCD (ms)');

for I := 0 to High(TestValues) do

begin

Writeln('Testing A = ', TestValues[I, 0], ', B = ', TestValues[I, 1]);

TotalTime := 0;

for J := 1 to RepeatCount do

begin

Stopwatch := TStopwatch.StartNew;

EuclidGCD(TestValues[I, 0], TestValues[I, 1]);

Stopwatch.Stop;

TotalTime := TotalTime + Stopwatch.ElapsedTicks;

end;

AvgTime := (TotalTime / RepeatCount) \* 1000 / Stopwatch.Frequency; // В миллисекундах

Writeln(TestValues[I, 0], #9, TestValues[I, 1], #9, FormatFloat('0.000', AvgTime));

end;

end;

begin

try

time;

except

on E: Exception do

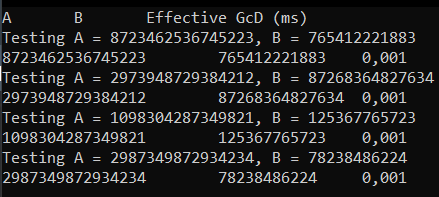
Writeln('Error: ', E.Message);

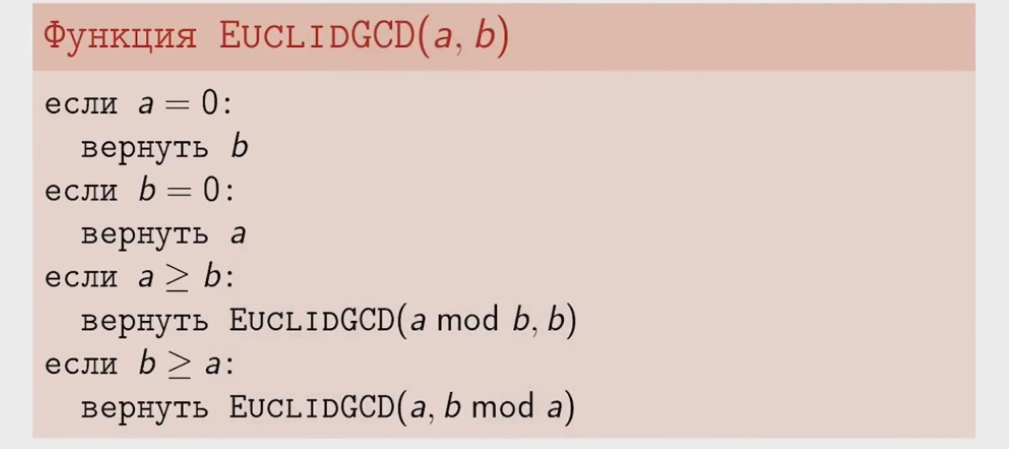
end;

Readln;

end.

**Время выполнения:**

****



**Вывод:**

На моей системе разницы в скорости выполнения программы различными способами не наблюдается, однако более эффективным будет 2-ой способ без рекурсии , т.к. будет выполняться меньшее количество строк кода.