



# **DESCRIPCIÓN, ANÁLISIS Y COMPARACIÓN DE DIVERSAS FORMULACIONES DEL ALGORITMO SIMPLE Y EXTENDIDO DE EUCLIDES**

Participantes:

- Eileen Karin Apaza Coaquira
- Renzo Leonardo Gallegos Vilca
- Xiomara Leonor Puma Torres
- Santiago Alonso San Roman Olazo



## Algoritmos

### Algoritmo de Euclides

1. Algoritmo de Euclides Básico
2. Algoritmo Binario del mcd
3. Algoritmo de Euclides con menor resto

### Algoritmo de Euclides Extendido

1. Algoritmo de Euclides Extendido Recursivo
2. Algoritmo de Euclides Extendido Iterativo.

# El mejor algoritmo de Euclides

## *Algoritmo de Euclides con menor resto*

### Concepto Matemático

Ceil( x ): Devuelve el menor entero mayor o igual a x.

$$\textcolor{blue}{x} = \text{Min } \{n \in \mathbb{Z} \mid n \geq x\}$$

Ejemplo:

$$\text{ceil}(2.25) = 3, \text{ceil}(2) = 2, \text{Ceil}(-2.25) = -2$$

Floor(x): Devuelve el más grande entero menor o igual a x.

$$\textcolor{red}{x} = \text{Max } \{n \in \mathbb{Z} \mid n \leq x\}$$

Ejemplo:

$$\text{Floor}(2.8) = 2, \text{Floor}(-2) = -2, \text{Floor}(-2.3) = -3$$

Notemos que  $\textcolor{blue}{x} = \textcolor{red}{x}$  si y sólo si x es entero, en otro caso  $\textcolor{blue}{x} = \textcolor{red}{x} + 1$ .

Usamos Ceil(x) y Floor(x), para poder encontrar el menor resto .

# CÓDIGO DEL ALGORITMO DE EUCLIDES CON MENOR RESTO

```
#include <iostream>
#include <NTL/ZZ.h>
```

```
using namespace std;
using namespace NTL;
```

```
ZZ mcdMenorResto(ZZ a, ZZ b){
    ZZ c,d,r;
    if(a==0){
        c=b;
    }
    else{
        c=a;
        d=b;
    }
    while(d<0 or d>0){
        r=c-d*(c/d+1/2);
        c=d;
        d=r;
    }
    return abs(c);
}
```

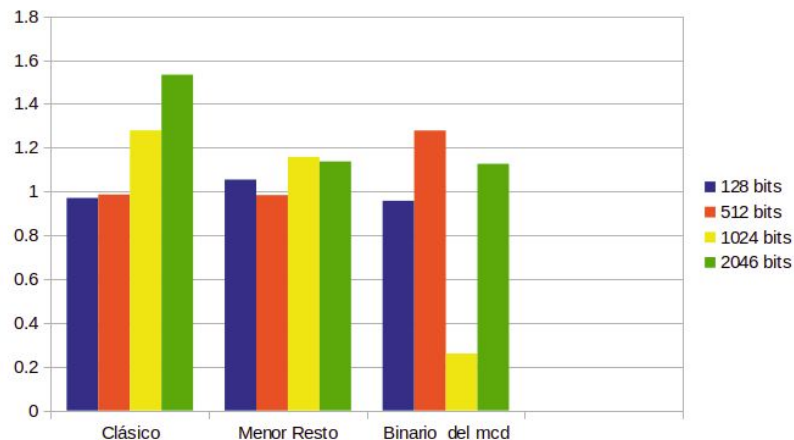
```
int main() {
    ZZ a,b;
    cin>>a;
    cin>>b;
    cout<<mcdMenorResto(a,b)<<endl;
    return 0;
}
```

# Comparación con el resto de algoritmos

Comparación del tiempo de ejecución según el N° de bits (128 – 512 – 1024 – 2046 ):

## A) Algoritmo de Euclides:

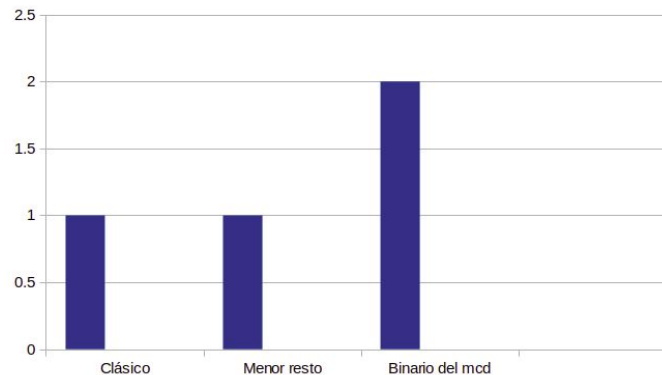
	128 bits	512 bits	1024 bits	2046 bits
Clásico	0.970	0.986	1.279	1.533
Menor Resto	1.054	0.983	1.158	1.137
Binario	0.957	1.278	1.260	1.126



Comparación del N° de loops:

## A) Algoritmo de Euclides:

	Clásico	Menor Resto	Binario
N° de loops	1	1	2



# El mejor algoritmo extendido de Euclides

Algoritmo extendido de Euclides iterativo

Concepto Matemático

Encuentra los coeficientes por los que el MCD de dos números se expresa en términos de los números mismos.

$x$	$y$	$(\text{rem}(x, y))$	$= x - q \cdot y$
259	70	49	$= 259 - 3 \cdot 70$
70	49	21	$= 70 - 1 \cdot 49$ $= 70 - 1 \cdot (259 - 3 \cdot 70)$ $= -1 \cdot 259 + 4 \cdot 70$
49	21	7	$= 49 - 2 \cdot 21$ $= (259 - 3 \cdot 70) - 2 \cdot (-1 \cdot 259 + 4 \cdot 70)$ $= \boxed{3 \cdot 259 - 11 \cdot 70}$
21	7	0	

## CÓDIGO EXTENDIDO DE EUCLIDES EN SU FORMA ITERATIVA

```
#include <iostream>
#include <NTL/ZZ.h>
```

```
using namespace std;
using namespace NTL;
```

```
void eucExt(ZZ a, ZZ b, ZZ& x, ZZ& y)
{
    x = ZZ(1), y = ZZ(0);

    ZZ x1(0), y1(1), a1(a), b1(b);
    while (b1 != 0)
    {
        ZZ q = a1 / b1;
        tie(x, x1) = make_tuple(x1, x - q * x1);
        tie(y, y1) = make_tuple(y1, y - q * y1);
        tie(a1, b1) = make_tuple(b1, a1 - q * b1);
    }
}
```

```
int main()
{
    ZZ x, y, a, b;
    cin>>a;
    cin>>b;
    eucExt(a, b, x, y);
    cout << "MCD(" << a << ", " << b << " ) = " <<
    eucExt(a, b, x, y) << endl;
    return 0;
}
```

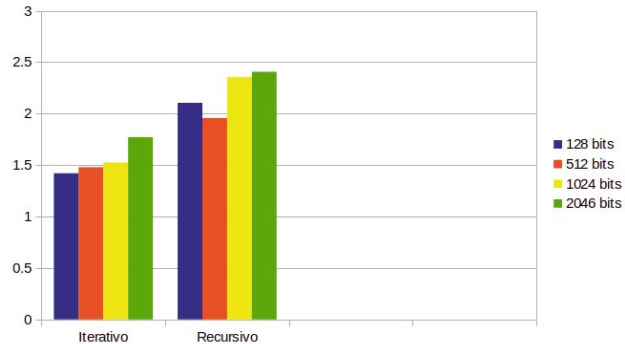
# Comparación con el resto de algoritmos



Comparación del tiempo de ejecución según el N° de bits (128 – 512 – 1024 – 2046 ):

## B) Algoritmo Extendido de Euclides:

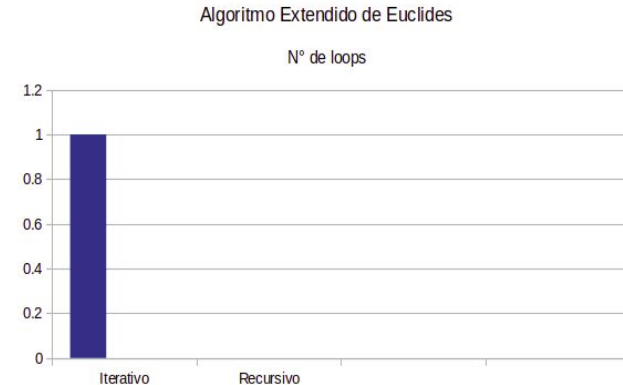
	128 bits	512 bits	1024 bits	2046 bits
Iterativo	1.420	1.479	1.526	1.771
Recursivo	2.105	1.957	2.355	2.407



Comparación del N° de loops:

## B) Algoritmo Extendido de Euclides:

	Iterativo	Recursivo
N° de loops	1	0







# Conclusiones

Evaluando todos los algoritmos de Euclides, llegamos a la conclusión que el mejor algoritmo es el "Algoritmo de euclides con menor resto" ya que es el más eficaz para el  $N^\circ$  de bits, tiene solo loop.

En el algoritmo extendido de euclides se llegó a la conclusión que el mejor algoritmo es el "Algoritmo extendido de Euclides en su forma iterativa" ya que no presenta demora al ejecutar el programa a comparación del recursivo es eficaz en  $N^\circ$  de bits.