ALGORITMOS
PROYECTO INTEGRADOR

NOMBRE: MATÍAS LLUMIQINGA-EMILIA CANO

EJERCICIO MULTIPLICACIÓN RUSA

Construir un algoritmo que permita multiplicar dos números enteros positivos empleando el método denominado MULTIPLICACIÓN RUSA. Este método permite calcular el producto de M*N de la siguiente manera:
En pasos sucesivos se divide M por 2 (división entera) y se multiplica N por 2. Este proceso se repite hasta que M es 0. El resultado de la multiplicación deseada se obtiene acumulando aquellos valores sucesivos de N para los cuales el valor de M es impar:

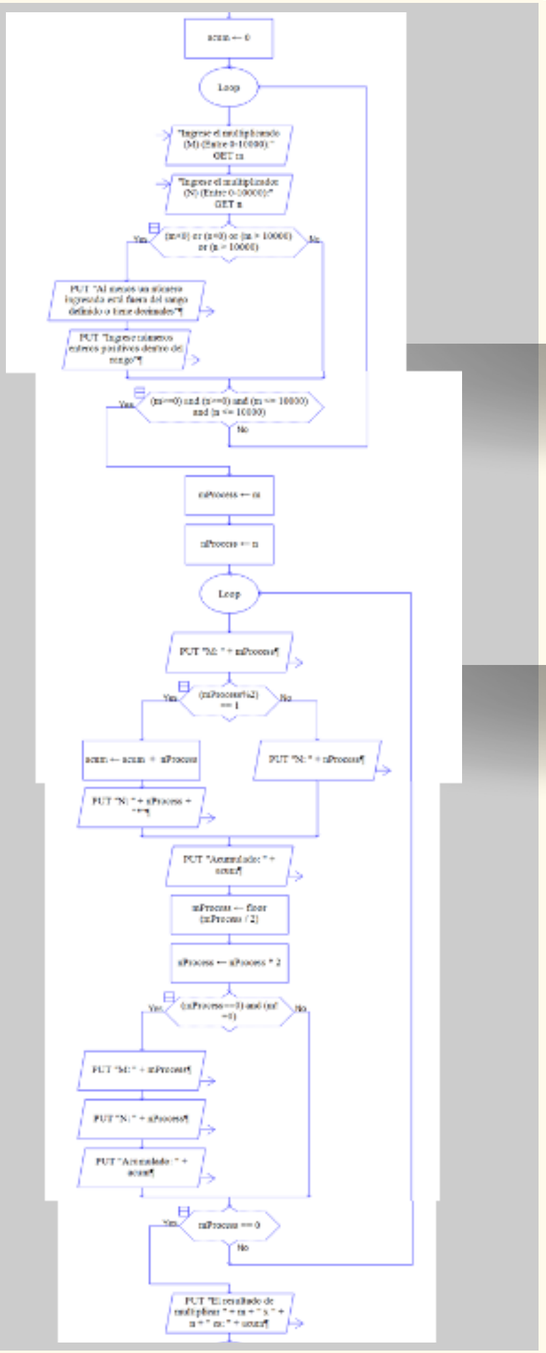
Primera solución
Análisis problema:

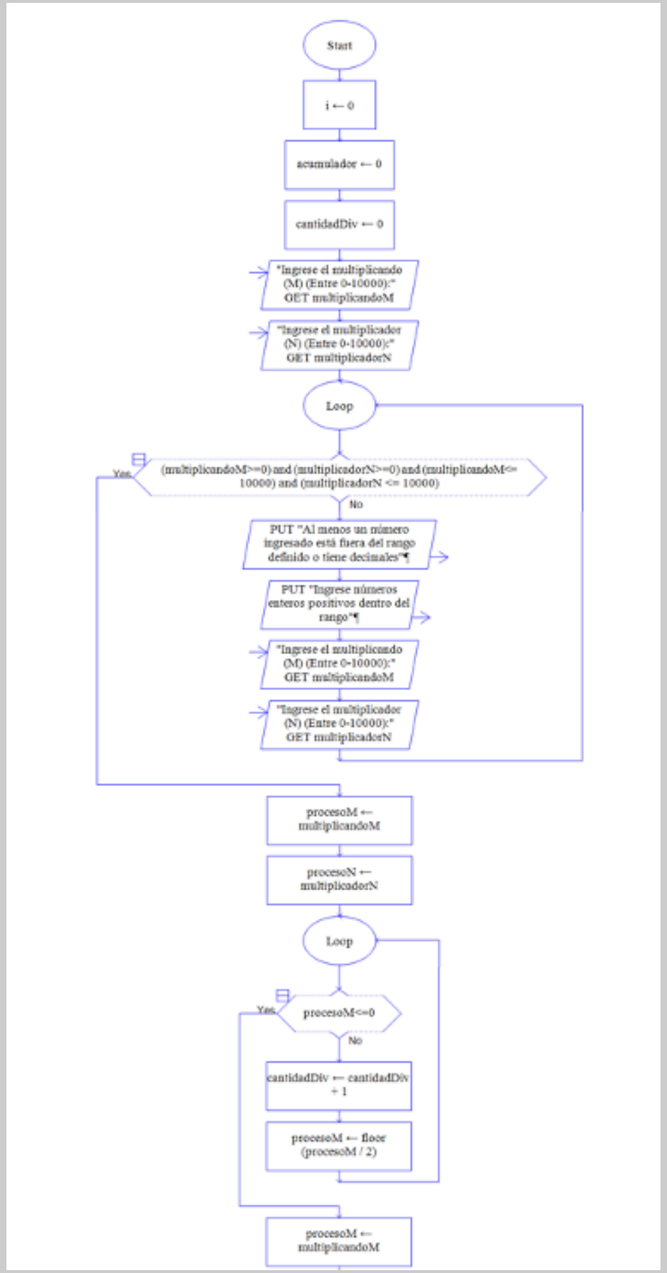
Entrada	Proceso	Salida
m		Valores de mProcess y nProcess y acum donde acum va sumandose a sí misma el valor de nProcess si mProcess es impar. Acum total.
n	acum = 0	
	Repetir	
	Leer m	
	Leer n	
	Si (m<0 o n<0 o m > 10000 o n > 10000) Entonces	
	Imprimir "Al menos un número ingresado está fuera del rango definido o tiene decimales"	
	Imprimir "Ingrese números enteros positivos dentro del rango"	
	Mientras (m<0 o n<0 o m > 10000 o n > 10000)	
	mProcess = m	
	nProcess = n	
	Repetir	
	Imprimir mProcess	
	Si (mProcess%2 == 1) Entonces	
	acum = acum + nProcess	
	Imprimir nProcess, "* "	
	Sino	
	Imprimir nProcess	
	mProcess = trunc(mProcess/2)	
	nProcess = nProcess* 2	
	Si (mProcess==0 y m<>0)	
	Imprimir mProcess, nProcess, acum	
	Mientras (mProcess>0)	
	Imprimir "total =", acum	

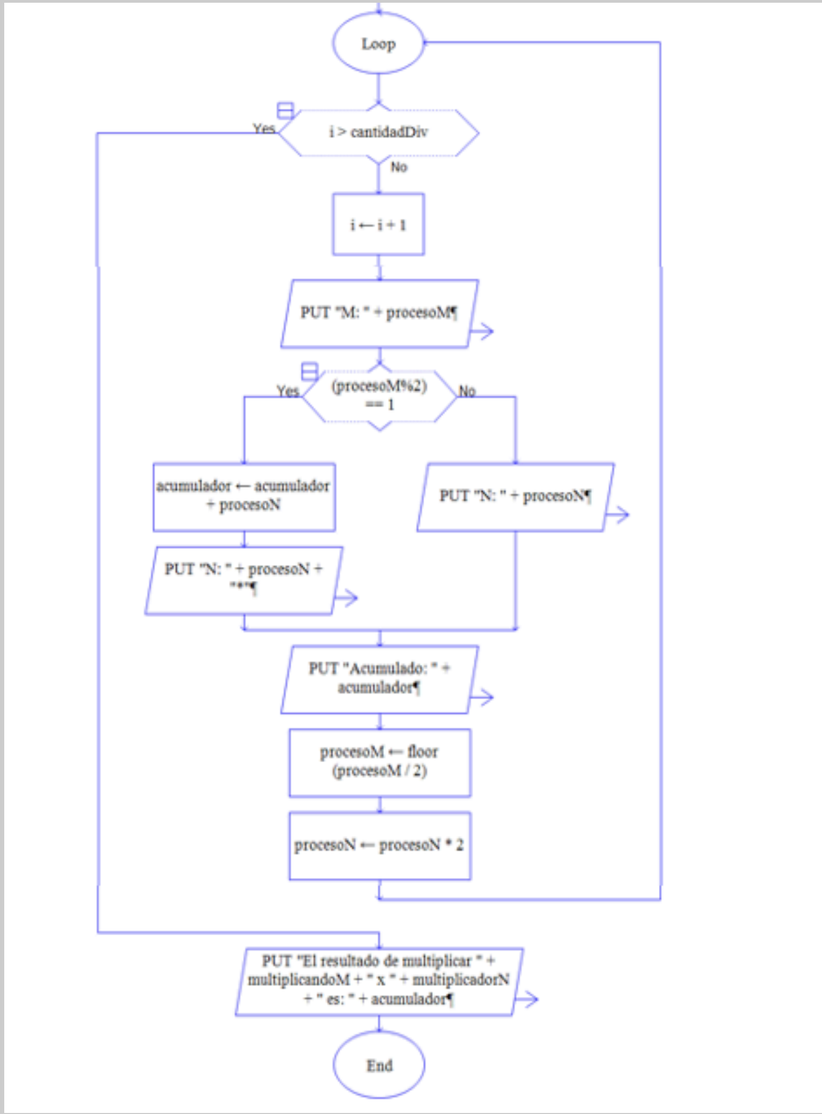
Segunda solución
Análisis problema

Entrada	Proceso	Salida
multiplicandoM		Valores de procesoM y procesoN y acumulador donde acumulador va sumandose a sí misma el valor de procesoN si procesoM es impar. Acumulador total.
multiplicadorN	acumulador = 0	
	cantidadDiv = 0	
	Leer multiplicandoM	
	Leer multiplicadorN	
	Segun (multiplicandoM<0 o multiplicadorN<0 o multiplicandoM > 10000 o multiplicadorN > 10000) Hacer	
	Imprimir "Al menos un número ingresado está fuera del rango definido o tiene decimales"	
	Imprimir "Ingrese números enteros positivos dentro del rango"	
	Leer multiplicandoM	
	Leer multiplicadorN	
	Fin Segun	
	procesoM = multiplicandoM	
	procesoN = multiplicadorN	
	Según (procesoM>0) Hacer	
	cantidadDiv = cantidadDiv+1	
	procesoM = trunc(procesoM/2)	
	Fin Segun	
	procesoM = multiplicandoM	
	Para i=0 Hasta cantidadDiv Con Paso 1 Hacer	
	Imprimir procesoM	
	Si (procesoM%2 == 1) Entonces	
	acumulador = acumulador + procesoN	
	Imprimir procesoN, "* "	
	Sino	
	Imprimir procesoN	
	Imrpimir acumulado	
	procesoM = trunc(procesoM/2)	
	procesoN = procesoN* 2	
	Fin Para	
	Imprimir "total =", acumulador	

DIAGRAMAS DE FLUJO







CÓDIGO EN C

```c
//#include <limits.h>    //printf("%i\n", INT_MAX);
#include <stdio.h>
#include <math.h>
#include <locale.h>
#include <conio.h>
#include <time.h>

int main(int argc, char const *argv[])
{
    setlocale(LC_CTYPE, "es_ES.UTF-8");

    int m, n, mProcess, nProcess, acum;
    clock_t start, end;
    double cpu_time_used;

    acum = 0;
    m = 0;
    n = 0;
    mProcess = 0;
    nProcess = 0;

    do
    {
        printf("Ingrese el multiplicando (M) (Entre 0-10000):\n");
        scanf("%i", &m);
        printf("Ingrese el multiplicador (N) (Entre 0-10000):\n");
        scanf("%i", &n);

        if (m<0 || n<0 || m>10000 || n>10000)
        {
            printf("Al menos un número ingresado está fuera del rango definido o tiene decimales\n");
            printf("Ingrese números enteros positivos dentro del rango\n");
        }
    } while (m<0 || n<0 || m>10000 || n>10000);

    //---------------------------------------START TIME----------------------------------------
    start = clock();
    //---------------------------------------END TIME----------------------------------------

    mProcess = m;
    nProcess = n;
    do
    {
        printf("M: %i\n", mProcess);

        if ((mProcess%2)==1)
        {
            acum = acum + nProcess;
            printf("N: %i*\n", nProcess);
        }
        else
        {
            printf("N: %i\n", nProcess);
        }

        printf("Acumulado: %i\n", acum);

        mProcess = trunc(mProcess/2);
        nProcess = nProcess*2;

        if (mProcess==0 && m!=0)
        {
            printf("M: %i\n", mProcess);        //Permite observar el caso en el que mProcess llega a 0
            printf("N: %i\n", nProcess);
            printf("Acumulado: %i\n", acum);
        }
    } while (mProcess > 0);

    printf("El resultado de multiplicar %i x %i, es: %i\n", m,n,acum);

    //---------------------------------------START TIME----------------------------------------
    end = clock();
    cpu_time_used = ((double) (end - start)) / CLOCKS_PER_SEC;

    printf("Tiempo de ejecución: %f\n", cpu_time_used);
    //---------------------------------------END TIME----------------------------------------

    getch();

    return 0;
}
```

PRUEBA DE ESCRITORIO:

Prueba de Escritorio														
Paso	acum	m	n	(m<0 || n<0 || m>10000 || n>10000)	(m<0 || n<0 || m>10000 || n>10000)	mProcess	nProcess	((mProcess %2) == 1)	acum	mProcess	nProcess	(mProcess==0 && m!=0)	(mProcess >0)	Salida
1	0													
2		30												
3			5											
4				FALSE										
5					FALSE									
6						30								
7							5							
8														"M: 30"
9								FALSE						
10									0					
11														"N: 5"
12														"Acumulado: 0"
13										15				
14											10			
15												FALSE		
16													TRUE	
17														"M: 15"
18								TRUE						
19									10					
20														"N: 10"
21														"Acumulado: 10"
22										7				
23											20			
24												FALSE		
25													TRUE	
26														"M: 7"
27								TRUE						
28									30					
29														"N: 20"
30														"Acumulado: 30"
31										3				
32											40			
33												FALSE		
34													TURE	
35														"M: 3"
36								TRUE						
37									70					
38														"N: 40"
39														"Acumulado: 70"
40										1				
41											80			
42												FALSE		
43													TRUE	
44														"M: 1"
45								TRUE						
46									150					
47														"N: 80"
48														"Acumulado: 150"
49										0				
50											160			
51												TRUE		
52														"M: 0"
53														"N: 160"
54														"Acumulado: 150"
55													FALSE	
														"El resultado de

CÓDIGO EN C

```c
#include <stdio.h>
#include <math.h>
#include <locale.h>
#include <conio.h>
#include <time.h>

int main(int argc, char const *argv[])
{
    setlocale(LC_CTYPE, "es_ES.UTF-8");

    int multiplicandoM, multiplicadorN, procesoM, procesoN, acumulador, cantidadDiv, i;
    clock_t start, end;
    double cpu_time_used;

    multiplicandoM = 0;
    multiplicadorN = 0;
    procesoM = 0;
    procesoN = 0;
    acumulador = 0;
    cantidadDiv = 0;

    printf("Ingrese el multiplicando (M) (Entre 0-10000):\n");
    scanf("%i", &multiplicandoM);
    printf("Ingrese el multiplicador (N) (Entre 0-10000):\n");
    scanf("%i", &multiplicadorN);

    while (multiplicandoM<0 || multiplicadorN<0 || multiplicandoM>10000 || multiplicadorN>10000)
    {
        printf("Al menos un número ingresado está fuera del rango definido o tiene decimales\n");
        printf("Ingrese números enteros positivos dentro del rango\n");

        printf("Ingrese el multiplicando (M) (Entre 0-10000):\n");
        scanf("%i", &multiplicandoM);
        printf("Ingrese el multiplicador (N) (Entre 0-10000):\n");
        scanf("%i", &multiplicadorN);
    }

    //-------------------------------------START TIME-------------------------------------
    start = clock();
    //-------------------------------------END TIME---------------------------------------
```

```c
    procesoM = multiplicandoM;
    procesoN = multiplicadorN;

    while (procesoM>0)
    {
        cantidadDiv = cantidadDiv+1;
        procesoM = trunc(procesoM/2);
    }

    procesoM = multiplicandoM;

    for (i = 0; i <= cantidadDiv; i++)
    {
        printf("M: %i\n", procesoM);

        if ((procesoM%2)==1)
        {
            acumulador = acumulador + procesoN;
            printf("N: %i*\n", procesoN);
        }
        else
        {
            printf("N: %i\n", procesoN);
        }

        printf("Acumulado: %i\n", acumulador);

        procesoM = trunc(procesoM/2);
        procesoN = procesoN*2;
    }

    printf("El resultado de multiplicar %i x %i, es: %i\n", multiplicandoM,multiplicadorN,acumulador);

    //-------------------------------------START TIME-------------------------------------
    end = clock();
    cpu_time_used = ((double) (end - start)) / CLOCKS_PER_SEC;

    printf("Tiempo de ejecución: %f\n", cpu_time_used);
    //-------------------------------------END TIME---------------------------------------

    getch();

    return 0;
}
```

PRUEBA DE ESCRITORIO

Prueba de escritorio

Paso	acum	cantidadDiv	multiplicandoM	multiplicadorN	(multiplicandoM <0 || multiplicadorN<0 || multiplicandoM> 10000 || multiplicadorN> 10000)	procesoM	procesoN	(procesoM>0)	cantidadDiv	procesoM	procesoM	(i = 0; i <= cantidadDiv; i++)		((mProcess % 2) == 1)	acumulador	procesoM	procesoN	Salida
1	0																	
2		0																
3			30															
4				5														
5					FALSE													
6																		
7						30												
8							5											
9								TRUE										
10									1									
11										15								
12								TRUE										
13									2									
14										7								
15								TRUE										
16									3									
17										3								
18								TRUE										
19									4									
20										1								
21								TRUE										
22									5									
23										0								
24								FALSE										
25											30							
26												TRUE	0					
27																		"M: 30"
28														FALSE				
29															0			
30																		"N: 5"
31																		"Acumulado: 0"
32																15		
33																	10	
34												TRUE	1					
35																		"M: 15"
36														TRUE				
37															10			
38																		"N: 15"
39																		"Acumulado: 10"
40																7		
41																	20	
42												TRUE	2					
43																		"M: 7"
44														TRUE				
45															30			
46																		"N: 20"
47																		"Acumulado: 30"
48																3		
49																	40	
50												TRUE	3					
51																		"M: 3"
52														TRUE				
53															70			
54																		"N: 40"
55																		"Acumulado: 70"
56																1		
57																	60	
58												TRUE	4					
59																		"M: 1"
60														TRUE				
61															100			
62																		"N: 60"
63																		"Acumulado: 100"
64																0		
65																	160	
66												TRUE	5					
67																		"M: 0"
68														FALSE				
69															150			
70																		"N: 150"
71																		"Acumulado: 150"
72																0		
73																	320	
74												FALSE	6					
75																		"El resultado de multiplicar 30 x 5 es: 150"

CUADRO COMPARATIVO

	Solución 1	Solución 2
Cantidad de variables	5	7
Tiempo de ejecución (Solo proceso)	Tiempo de ejecución: 0.014000	Tiempo de ejecución: 0.015000
Tiempo de ejecución (Todo el código)	Tiempo de ejecución: 5.345000	Tiempo de ejecución: 8.060000
Cantidad de líneas del código	84	87
Claridad en nombre de variables	Variables no tan descriptivas	Variables descriptivas

Se escoge la solución 1 frente a la solución 2, por que la 1 tiene más puntos positivos que benefician a la ejecución del código.

Conclusiones:

En conclusión, se elige la solución 1 frente a la 2, debido a que en la solución 1 la cantidad de variables es menor. Asimismo, el tiempo de ejecución y la cantidad de líneas de código es menor frente a la segunda solución. A pesar de ello, se puede observar una mayor claridad en el nombre de variables del segundo código. Es por ello, que, si se observa en el código, las variables son más específicas y descriptivas.