

**Nombre: Velázquez Ramírez Carlos Raúl**

**No. 426096993**

**Tema: Limitaciones de la computadora**

**Grupo: 1153**

**Fecha: 14/10/2025**

1. Utiliza una hoja para crear la siguiente tabla:

	A	B	C	D	E	F	G
1							
2	n	$2^n$	$(2^n)+1$	$[(2^n)+1]-1$	Nomenclatura binaria		
3		1	2	3	1		1
4		2	4	5	1		2
5		3	8	9	1		4
6		4	16	17	1		8
7		5	32	33	1		16
8		6	64	65	1		32
9		7	128	129	1		64
10		8	256	257	1		128
11		9	512	513	1		256
12		10	1024	1025	1		512
13		11	2048	2049	1 1k		
14		12	4096	4097	1		2
15		13	8192	8193	1		4
16		14	16384	16385	1		8
17		15	32768	32769	1		16
18		16	65536	65537	1		32
19		17	131072	131073	1		64
20		18	262144	262145	1		128
21		19	524288	524289	1		256
22		20	1048576	1048577	1		512
23		21	2097152	2097153	1 1M		
24		22	4194304	4194305	1		2
25		23	8388608	8388609	1		4
26		24	16777216	16777217	1		8
27		25	33554432	33554433	1		16
28		26	67108864	67108865	1		32
29		27	134217728	134217729	1		64
30		28	268435456	268435457	1		128
31		29	536870912	536870913	1		256
32		30	1073741824	1073741825	1		512
33		31	2147483648	2147483649	1 1G		
34		32	4294967296	4294967297	1		2
35		33	8589934592	8589934593	1		4
36		34	17179869184	17179869185	1		8
37		35	34359738368	34359738369	1		16
38		36	68719476736	68719476737	1		32
39		37	1.37439E+11	1.37439E+11	1		64
40		38	2.74878E+11	2.74878E+11	1		128
41		39	5.49756E+11	5.49756E+11	1		256
42		40	1.09951E+12	1.09951E+12	1		512
43		41	2.19902E+12	2.19902E+12	1 1T		
44		42	4.39805E+12	4.39805E+12	1		2
45		43	8.79609E+12	8.79609E+12	1		4
46		44	1.75922E+13	1.75922E+13	1		8
47		45	3.51844E+13	3.51844E+13	1		16
48		46	7.03687E+13	7.03687E+13	1		32

	A	B	C	D	E	F	G
18		16	65536	65537	1	32	
19		17	131072	131073	1	64	
20		18	262144	262145	1	128	
21		19	524288	524289	1	256	
22		20	1048576	1048577	1	512	
23		21	2097152	2097153	11M		
24		22	4194304	4194305	1	2	
25		23	8388608	8388609	1	4	
26		24	16777216	16777217	1	8	
27		25	33554432	33554433	1	16	
28		26	67108864	67108865	1	32	
29		27	134217728	134217729	1	64	
30		28	268435456	268435457	1	128	
31		29	536870912	536870913	1	256	
32		30	1073741824	1073741825	1	512	
33		31	2147483648	2147483649	11G		
34		32	4294967296	4294967297	1	2	
35		33	8589934592	8589934593	1	4	
36		34	17179869184	17179869185	1	8	
37		35	34359738368	34359738369	1	16	
38		36	68719476736	68719476737	1	32	
39		37	1.37439E+11	1.37439E+11	1	64	
40		38	2.74878E+11	2.74878E+11	1	128	
41		39	5.49756E+11	5.49756E+11	1	256	
42		40	1.09951E+12	1.09951E+12	1	512	
43		41	2.19902E+12	2.19902E+12	11T		
44		42	4.39805E+12	4.39805E+12	1	2	
45		43	8.79609E+12	8.79609E+12	1	4	
46		44	1.75922E+13	1.75922E+13	1	8	
47		45	3.51844E+13	3.51844E+13	1	16	
48		46	7.03687E+13	7.03687E+13	1	32	
49		47	1.40737E+14	1.40737E+14	1	64	
50		48	2.81475E+14	2.81475E+14	1	128	
51		49	5.6295E+14	5.6295E+14	1	256	
52		50	1.1259E+15	1.1259E+15	1	512	
53		51	2.2518E+15	2.2518E+15	11P		
54		52	4.5036E+15	4.5036E+15	1	2	
55		53	9.0072E+15	9.0072E+15	0	4	
56		54	1.80144E+16	1.80144E+16	0	8	
57		55	3.60288E+16	3.60288E+16	0	16	
58		56	7.20576E+16	7.20576E+16	0	32	
59		57	1.44115E+17	1.44115E+17	0	64	
60		58	2.8823E+17	2.8823E+17	0	128	
61		59	5.76461E+17	5.76461E+17	0	256	
62		60	1.15292E+18	1.15292E+18	0	512	
63							

2. Haz la prueba en tu computadora y verifica en qué momento la diferencia cambia de uno a cero:

El momento donde el valor pasa a ser 0 es en 4 petabytes.

3. ¿El cambio sucede con los mismos valores de la imagen?

No, sucede incluso antes.

4. ¿A qué atribuyes ese cambio de comportamiento?

Probablemente una cuestión de capacidad o “ajuste fino” de la computadora. Computadoras menos capaces probablemente empiezan a fallar en ciertas operaciones que computadoras más capaces sí pueden,

5. Investiga y explica por qué la computadora no se está comportando de acuerdo a lo esperado (que la diferencia sea siempre de uno):

Según Claude.ai, resumido por mí:

Básicamente es el límite de los 64 bits en double.

Algunos programas compilan/trabajan en double, y el límite de 64 bits en double se encuentra, justamente, en  $2^{53}$ , añadir más resulta en fallos, pues la computadora no está diseñada para eso, es una cuestión de Hardware, de capacidad.

Y la razón por la que estos son diferentes se debe a que el límite varía según los bits de la computadora (e.g. el límite de 32 bits está en  $2^{24}$ ).

6. Considera el siguiente código en C:

vela@velasthinkpad:~/Documents

```

odc.c  x
#include <math.h>
#include <stdio.h>
int main()
{
    int n;
    for (n=1; n<=60; n++) {
        float result = pow(2, n);
        float resulta = result + 1;
        printf("2^%d = %.0lf +1= %.0lf Dif: %.0lf\n", n, result, resulta, (resulta - result));
    }
    return 0;
}

```

vela@velasthinkpad:~/Documents

```

2^15 = 32768 +1= 32769 Dif: 1
2^16 = 65536 +1= 65537 Dif: 1
2^17 = 131072 +1= 131073 Dif: 1
2^18 = 262144 +1= 262145 Dif: 1
2^19 = 524288 +1= 524289 Dif: 1
2^20 = 1048576 +1= 1048577 Dif: 1
2^21 = 2097152 +1= 2097153 Dif: 1
2^22 = 4194304 +1= 4194305 Dif: 1
2^23 = 8388608 +1= 8388609 Dif: 1
2^24 = 16777216 +1= 16777216 Dif: 0
2^25 = 33554432 +1= 33554432 Dif: 0
2^26 = 67108864 +1= 67108864 Dif: 0
2^27 = 134217728 +1= 134217728 Dif: 0
2^28 = 268435456 +1= 268435456 Dif: 0
2^29 = 536870912 +1= 536870912 Dif: 0
2^30 = 1073741824 +1= 1073741824 Dif: 0
2^31 = 2147483648 +1= 2147483648 Dif: 0
2^32 = 4294967296 +1= 4294967296 Dif: 0
2^33 = 8589934592 +1= 8589934592 Dif: 0
2^34 = 17179869184 +1= 17179869184 Dif: 0
2^35 = 34359738368 +1= 34359738368 Dif: 0
2^36 = 68719476736 +1= 68719476736 Dif: 0
2^37 = 137438953472 +1= 137438953472 Dif: 0
2^38 = 274877906944 +1= 274877906944 Dif: 0
2^39 = 549755813888 +1= 549755813888 Dif: 0
2^40 = 1099511627776 +1= 1099511627776 Dif: 0
2^41 = 2199023255552 +1= 2199023255552 Dif: 0
2^42 = 4398046511104 +1= 4398046511104 Dif: 0
2^43 = 8796093022208 +1= 8796093022208 Dif: 0
2^44 = 17592186044416 +1= 17592186044416 Dif: 0

```

NORMAL odc.c

LSP ~ clangd Documents 12/1

7. ¿Los valores en los que la diferencia cambia son los mismos en ambos casos (hoja de cálculo y computadora)?

Sí, pasan a 0 en  $2^{24}$ .

8. Sustituye en el programa el tipo de las variables (“result” y “resulta”) por doble precisión (“double”). ¿Qué cambios observas en los resultados?

The screenshot shows a terminal window with two panes. The left pane displays a file tree of documents and source code files. The right pane shows the output of a C program named odc.c.

```
vela@velasthinkpad: ~/Documents
```

```
odc.c  x
#include <math.h>
#include <stdio.h>
int main()
{
    int n;
    for (n=1; n<=60; n++) {
        double result = pow(2, n);
        double resulta = result + 1;
        printf("2^%d = %.0lf +1= %.0lf Dif: %.0lf\n", n, result, resulta, (resulta - result));
    }
    return 0;
}
```

```
vela@velasthinkpad: ~/Documents
```

```
2^32 = 4294967296 +1= 4294967297 Dif: 1
2^33 = 8589934592 +1= 8589934593 Dif: 1
2^34 = 17179869184 +1= 17179869185 Dif: 1
2^35 = 34359738368 +1= 34359738369 Dif: 1
2^36 = 68719476736 +1= 68719476737 Dif: 1
2^37 = 137438953472 +1= 137438953473 Dif: 1
2^38 = 274877986944 +1= 274877986945 Dif: 1
2^39 = 549755813888 +1= 549755813889 Dif: 1
2^40 = 1099511627776 +1= 1099511627777 Dif: 1
2^41 = 2199023255552 +1= 2199023255553 Dif: 1
2^42 = 4398046511104 +1= 4398046511105 Dif: 1
2^43 = 8796093022208 +1= 8796093022209 Dif: 1
2^44 = 17592186044416 +1= 17592186044417 Dif: 1
2^45 = 35184372088832 +1= 35184372088833 Dif: 1
2^46 = 70368744177664 +1= 70368744177665 Dif: 1
2^47 = 140737488355328 +1= 140737488355329 Dif: 1
2^48 = 281474976710656 +1= 281474976710657 Dif: 1
2^49 = 562949953421312 +1= 562949953421313 Dif: 1
2^50 = 1125899906842624 +1= 1125899906842625 Dif: 1
2^51 = 2251799813685248 +1= 2251799813685249 Dif: 1
2^52 = 4503599627370496 +1= 4503599627370497 Dif: 1
2^53 = 9007199254740992 +1= 9007199254740992 Dif: 0
2^54 = 18014398509481984 +1= 18014398509481984 Dif: 0
2^55 = 36028797018963968 +1= 36028797018963968 Dif: 0
2^56 = 72057594037927936 +1= 72057594037927936 Dif: 0
2^57 = 144115188075855872 +1= 144115188075855872 Dif: 0
2^58 = 288230376151711744 +1= 288230376151711744 Dif: 0
2^59 = 576460752303423488 +1= 576460752303423488 Dif: 0
2^60 = 1152921504606846976 +1= 1152921504606846976 Dif: 0
```

```
vela@velasthinkpad: ~/Documents
```

```
NORMAL  odc.c
"odc.c" 12L, 258B written
```

```
LSP ~ clangd  Documents  8/11
```

El límite pasa a ser  $2^{53}$ , que es justamente el mismo que en la hoja de cálculo.

9. ¿Cómo se justifica el comportamiento de la computadora?

Se debe a la conducta del tipo de dato. Float trabaja sobre una línea de memoria de bits limitados, cuyo límite se encuentra justamente a  $2^{24}$ . Double, o doble presición/punto flotante, da uso de memoria más allá de la línea de memoria asignada, sobrepasando el límite de  $2^{24}$  y llegando al monumental número de  $2^{53}$ .

10. ¿Cuáles son tus conclusiones de este ejercicio?

Aunque no creo necesitarlo inmediatamente, tendré que considerarlo siempre al programar o lidiar con software que de uso de cálculos enormes. Como quien pretende ser matemático, esto es de muchísima utilidad.