Representación de la información
Ejercicios resueltos

Ejercicio 1. Indique la representación de los siguientes números, razonando su respuesta:

a) -16 en complemento a 2 con 5 bits
b) -16 en complemento a 1 con 5 bits
c) +13 en signo magnitud con 5 bits
d) -14 en complemento a dos con 5 bits

Solución:

a) El rango de representación de números en complemento a dos con 5 bits es [-25-1..25-1-1] = [-16..15]. 16 en binario es 10000. Tenemos que complementar, 01111 y sumarle 1. Por tanto, -16 en complemento a dos con 5 bits es 10000.
b) El rango de representación de números en complemento a dos con 5 bits es [-25-1..25-1-1] = [-15..15]. Por tanto, el número -16 no se puede representar.
c) 13 en binario puro es 1101. En signo magnitud se introduce al comienzo un bit de signo, en este caso 0 para indicar que es positivo. El número 13 es signo magnitud con 5 bits es 01101.
d) 14 en binario puro con 5 bits es 01110. Se complementa, 10001 y se suma 1, con lo que se obtiene 10010.

Ejercicio 2. Indique la representación de los siguientes números:

a) -64 en complemento a uno con 7 bits
b) -64 en complemento a dos con 7 bits
c) 12 en signo magnitud con 6 bits
d) 18 en complemento a dos con 5 bits

Solución:

a) -64 en complemento a uno con 7 bits
El rango de representación es [-63, 63], por tanto, el número no es representable
b) -64 en complemento a dos con 7 bits
64 en binario con 7 bits es 1000000. Se complementa 0111111 y se suma 1, siendo el resultado 100000
c) 12 en signo magnitud con 6 bits
001100
d) 18 en complemento a dos con 5 bits
El rango de representación es [-16, 15], por tanto, el número no es representable.

Ejercicio 3. ¿Cómo se detecta un desbordamiento en Complemento a 2 cuando se hace una operación de suma?

Solucón:

Se detecta cuando los dos operandos tienen el mismo signo y el resultado tiene signo distinto al de los operandos.

Ejercicio 4. Represente en el estándar IEEE 754 de simple precisión el valor -36.

Solución:

El valor 36 en binario es 100100. 100100 = 1.00100 x 2^5. Por tanto:

- El bit de signo es 1, porque el número es negativo.
- El exponente es 5, por tanto el exponente que se almacena es 5 + 127 = 132, que en binario es 10000100
- La mantisa es 001000000 …. 00000

Por tanto, el número -36 se representa como 11000010000100000000000000000000 en binario.

Ejercicio 5. Indique el valor decimal del siguiente número hexadecimal 0x00600000 que representa un número en coma flotante según IEEE 754 (precisión simple)

Solución:

0x00600000 en binario es 00000000011000000000000000000000

Signo = 0, número positivo
Exponente = 00000000
Mantisa = 1100000…0000

Se trata de un número no normalizado cuyo valor es $0,11 \times 2^{-126} = 0,75 \times 2^{-126}$

Ejercicio 6. Represente el número -24,50 utilizando el estándar de coma flotante de simple precisión IEEE 754. Exprese dicha representación en binario y en hexadecimal.

Solución:

$24,5_{(10} = 11000.1_{(2} = 1,10001 \times 2^{4}$

Signo = 1, número negativo
Exponente = 4 + 127 = 131 = 10000011
Mantisa = 1000100000 .. 00000

En binario es: 11000000111000100000 …….00000
En Hexadecimal es: 0xC1C40000

Ejercicio 7. Se desea representar números enteros dentro del rango -8191…8191. Indicar de forma razonada:

a) ¿Cuál es el número de bits que se necesita si se quiere utilizar una representación en complemento a uno?
b) ¿Cuál es el número de bits que se necesita si se quiere utilizar una representación en signo-magnitud?

Solución:

Se necesitan en los dos casos 14 bits. Con 14 bits el rango de representación en ambos casos es de
$-(2^{13}-1) \ldots 2^{13}-1 = -8191 \ldots 8191$

Ejercicio 8. ¿Cuál es el número positivo normalizado más pequeño que se puede representar utilizando el estándar de simple precisión IEEE 754? Justifique su respuesta. Indique también el número positivo no normalizado más pequeño que se puede representar. Justifique de igual forma su respuesta.

Solución:

a) El número positivo normalizado más pequeño representable en el estándar IEEE 754 (32 bits) es:
   0 00000001 00000000000000000000000
   positivo normalizado más pequeño

   Cuyo valor es $1.0 * 2^{1-127} = \mathbf{2^{-126}}$

b) El número positivo no normalizado más pequeño representable en el estándar es:
   0 00000000 00000000000000000000001
   positivo no normalizado más pequeño

   Cuyo valor es $2^{-23} * 2^{-126} = \mathbf{2^{-149}}$

Ejercicio 9. Represente en el estándar de coma flotante IEEE 754 de 32 bits los valores 10,25 y 6,75. Exprese el resultado final en hexadecimal. Realice, a continuación, la suma de los números anteriores representados en IEEE 754, indicando los pasos que va realizando en cada momento.

Solución:

a) $10,25_{(10} = 1010,01_{(2} = 1.01001 \times 2\text{^}3_{(2}$
   Signo = 0

Exponente = 127+3 = $130_{(10} = 10000010_{(2}$
Mantisa = 010010000…00

En Binario: 0100 0001 0010 0100 00…00
En Hexadecimal: 0x41240000

$6{,}75_{(10} = 110{,}11_{(2} = 1.1011 \times 2\text{^}2_{(2}$
Signo = 0
Exponente = 127+2 = $129_{(10} = 10000001_{(2}$
Mantisa = 10110000…00

En Binario: 0100 0000 1101 1000 00…00
En Hexadecimal: 0x40D80000

b) Para sumar ambos números en IEEE754, lo primero que tenemos que hacer es igualar exponentes, y tener en cuenta a la hora de sumar el bit implícito de la mantisa, después realizar la suma, y si el resultado no está normalizado, normalizarlo.

10,25 = 0 10000010 1.01001000000000000000000
6,75 = 0 10000010 0.11011000000000000000000
Suma = 0 10000010 10.00100000000000000000000
Resultado normalizado = 0 10000011 00010000000000000000000
En Hexadecimal: 0x41880000

Ejercicio 10. Indique el valor decimal del siguiente número representado en el estándar IEEE 754 de simple precisión: 0xBF400000.

Solución:

El valor decimal de 0xBF400000 es el siguiente:

En Binario: 1011 1111 0100 0000…00

Signo= Negativo
Exponente = $01111110_{(2} = 126_{(10}$ = Exponente = -1
Mantisa = 1

Número en binario: $-1{,}1 \times 2^{-1}{}_{(2} = -0{,}11_{(2} = -0{,}75(10$

Ejercicio 11. En relación al estándar IEEE 754 responda a las siguientes preguntas:

a) En una representación IEEE 754 de 32 bits, indique de forma razonada el número de valores no normalizados que se pueden representar.
b) En un computador de 32 bits, ¿Se puede representar de forma exacta el valor $2^{27}+1$ en una variable de tipo `float`? ¿y en una variable de tipo `int`? Razone su respuesta.
c) Represente en el estándar IEEE 754 de doble precisión el valor 12.5. Exprese el resultado en hexadecimal.

Solución:

a) Un valor no normalizado se corresponde con un exponente (8 bits) cuyo valor es 0 y una mantisa (23 bits) con valor distinto de cero. El número de elementos que se pueden representar es, por tanto, 2 (positivo y negativo) x $2^{23}-1 = 2 \cdot (2^{23}-1)$

b) El valor $2^{27} + 1 = 1000000000000000000000000001_{(2} = 1.000000000000000000000000001 \times 2^{27}{}_{(2}$

Dado que en la mantisa de un número de tipo `float` (IEEE 754 de 32 bits) solo se pueden almacenar 23 bits, aparte del bit implícito, para el número anterior no se podrían almacenar los 4 bits menos significativos del número, por lo que el número no se podría representar de forma exacta. En una variable de tipo `int`, que emplea complemento a 2, el rango de representación es $[-2^{31}, 2^{31}-1$. En este caso, sí que se puede representar el número de forma exacta.

c)

$12.5_{(10} = 1100.1_{(2} = 1.1001 \times 2^3{}_{(2}$

Signo = 1
Exponente = 1023+3 = $1026_{(10}$ $1024+2_{(10}$ = $10000000010_{(2}$
Mantisa = 1001…. 000000 (52 bits)

La representación del valor 12.5 es:

0 100 0000 0010 1001 00000000 …… 00000

Expresado en hexadecimal queda: 0x4029000000000000

Ejercicio 12. Considere el siguiente fragmento de programa escrito en C, que ejecuta en un computador de 32 bits.

```
double A;
float  B;
int     C;

A =  pow(2, 28) + 5;   // 2^28 +5
B = (float) A;
C = (int)   A;
```

Después de ejecutar el fragmento de código anterior, indique de forma razonada el valor, en hexadecimal, que se encuentra almacenado en las variables A, B y C.

Solución:

A = pow(2, 28) + 5; // 2^{28} +5

El valor $2^{28} + 5 = 10000000000000000000000000101_{(2}$
$= 1.0000000000000000000000000101 \times 2^{28}{}_{(2}$ de forma normalizada.

Cuando se almacena este valor en una variable de tipo `double`, que se corresponde con el formato IEEE 754 de doble precisión se obtiene:
Signo = 0
Exponente = 1023+28 = $1051_{(10}$ $1024+16+11_{(10}$ = $10000011011_{(2}$
Mantisa = 0000000000000000000000000101000000000 ….. 000000 (52 bits)

El valor almacenado es en binario:

0 10000011011 0000000000000000000000000101000000000 ….. 000000

y en hexadecimal: 0x41B0000005000000

B = (float) A;

El valor $2^{28} + 5 = 1.0000000000000000000000000101 \times 2^{28}{}_{(2}$ de forma normalizada.

Cuando se almacena este valor en una variable de tipo `float`, que se corresponde con el formato IEEE 754 de simple precisión se obtiene:
Signo = 0
Exponente = 127+28 = $155_{(10}$ = $10011011_{(2}$
Mantisa = 0000…. 000 (23 bits, se pierden los últimos bits menos significativos)

El valor almacenado es en binario:

0 10011011 0000…. 000

y en hexadecimal: 0x4D80000000000000

La mantisa de un número de tipo `float` (IEEE 754 de 32 bits) solo se pueden almacenar 23 bits (bit implícito aparte) por lo que no se podrían almacenar los 4 bits menos significativos del número.

C = (int) A;

Cuando se almacena el valor de A (de tipo double) $10000000000000000000000000101_{(2}$ en una variable de 32 bits en complemento a dos se obtiene:

0001 0000 0000 0000 0000 0000 0000 0101

y en hexadecimal: 0x10000005

En una variable de tipo `int`, que emplea complemento a 2, el rango de representación si permite representar el número de forma exacta este valor.

Ejercicio 13. En relación al estándar IEEE 754 responda a las siguientes preguntas:

a) En una representación IEEE 754 de 32 bits, ¿cuál es el número de valores que hay comprendidos entre el número 8 y el 9?
d) ¿Se mantiene constante, en este estándar, el número de valores representables para cualquier intervalo [N, N+1] siendo N un valor entero? Razone su respuesta.
e) ¿Dónde tiene más precisión una variable de tipo *float*, en un computador de 32 bits o en uno de 64 bits? Razone su respuesta.
f) ¿Indique de forma razonada y justificada si se puede almacenar de forma exacta cualquier entero de 32 bits en complemento a dos en una variable de tipo *float*?

Solución:

a) Para poder calcular el número de valores comprendidos entre el 8 y el 9 es necesario representar ambos números en coma flotante:

$8_{(10} = 1000_{(2} = 1.000 \times 2^3{}_{(2}$

Signo = 1
Exponente = 127+3 = $130_{(10} = 10000010_{(2}$
Mantisa = 000000…. 000000 (23 bits)

$9_{(10} = 1001_{(2} = 1.001 \times 2^3{}_{(2}$

Signo = 1
Exponente = 127+3 = $130_{(10} = 10000010_{(2}$
Mantisa = 001000…. 000000 (23 bits)

La representación del valor 8 es: 0 10000010 00000000000000000000000
La representación del valor 9 es: 0 10000010 00100000000000000000000

Lo único que hay que hacer es calcular el número de valores comprendidos entre ambas representaciones. El primer bit (comenzando desde la derecha) con un valor distinto es el que se encuentra en la posición 21, por tanto el número de valores es 2^{20}.

b) Este valor no se mantiene constante para cualquier intervalo, puesto que la resolución de la representación, es decir, la diferencia entre dos valores representables, no se mantiene constante para el estándar IEEE 754.

c) Una variable de tipo *float* utiliza el estándar IEEE 754 de 32 bits, por tanto, su resolución no depende del ancho de palabra del computador utilizado.

d) No se puede representar de forma exacta cualquier valor de tipo entero. Considere el valor $2^{30} + 5$. Este valor se puede almacenar en una variable entera de 32 bits en complemento a dos, puesto que el rango de representación es $[-2^{31}, 2^{31} -1]$. Cuando se almacena el valor $2^{30} + 5$ en una variable de tipo *float* se obtiene lo siguiente:

$2^{30} + 5 = 1000000000000000000000000000101_{(2} = 1.000000000000000000000000000101 \times 2^{30}{}_{(2}$

En la mantisa solo se pueden almacenar los 23 primeros bits de la parte fraccionaria, por lo que se pierden los bits menos significativos.

Ejercicio 14. En relación al estándar IEEE 754 responda a las siguientes preguntas:

a) Indique el contenido en bits de una variable de tipo `double` cuando se almacena en ella el valor 17.25.
b) ¿Qué ocurre cuando se almacena una variable de tipo `double` en una variable de tipo `float`?

Solución:

a) Una variable de tipo `double` en formato IEEE 754 ocupa 64 bits, y está formado por 1 bit para el signo, 11 para el exponente y 52 para la mantisa (en lugar de los 1, 8 y 23 bits respectivamente del formato de 32 bits).

$17.25_{10} = 16 + 1 + 0.25 = 2^4 + 2^0 + 2^{-2} = 10001.01_{(2}$
$10001.01_{(2} = 1.000101_{(2} * 2^4$

Signo: 0 (el número es positivo)
Exponente real= 4; el Exponente que se almacena es $4+ 2^{11}-1 = 4 + 2047 = 2051_{10} = 10000000011_{(2}$
Mantisa: $000101000000\ldots..00000000_{(2}$

En hexadecimal, 17.25 en IEEE 754 es: 4031400000000000

b) Que hay que adaptar la representación del número, pasando de 64 bits a 32 bits. Es posible que:
- El número no sea representable (al ser el rango mayor) .
- que se pierda precisión (al dedicar menos bits para la mantisa).

NOTA: Puede verse el resultado en http://babbage.cs.qc.edu/IEEE-754/

Ejercicio 15. En relación al estándar IEEE 754 responda a las siguientes preguntas:

a) En una representación IEEE 754 de 32 bits, indique de forma razonada el número de valores no normalizados que se pueden representar.
b) Represente en el estándar IEEE 754 de doble precisión el valor -20.5. Exprese el resultado en hexadecimal.

Solución:

a) Un valor no normalizado se corresponde con un exponente (8 bits) cuyo valor es 0 y una mantisa (23 bits) con valor distinto de cero. El número de elementos que se pueden representar es, por tanto, 2 (positivo y negativo) x 223-1 = 2(223-1)

b) -20.5(10 = 10100.1(2 = 1.01001 x 24(2

Signo = 1
Exponente = 1023+4 = 1027(10 1024+3(10 = 10000000011(2
Mantisa = 01001…. 000000 (52 bits)

La representación del valor -20.5 es:

1 100 0000 0011 0100 1000000…….000000

Expresado en hexadecimal queda: 0xC034800000000000

Ejercicio 16. En relación al estándar IEEE 754 responda a las siguientes preguntas:

a) Dado el número 0.6, ¿en cuál de los formatos (simple precisión o doble precisión) se puede representar el número 0.6 de forma exacta. Razone su respuesta.
b) Represente en el estándar IEEE 754 de doble precisión el valor -18.25. Exprese el resultado en hexadecimal.

Solución:

a) El número 0.6 no se puede representar de forma exacta en binario:

$0.6_{(10} = 0,100110011001\ldots.$

Por tanto, no se puede representar de forma exacta ni en simple ni en doble precisión.

b) -18.25(10 = 10010.01(2 = 1.001001 x 24(2

Signo = 1
Exponente = 1023+4 = 1027(10 1024+3(10 = 10000000011(2
Mantisa = 00100100000…..0000000 (52 bits)

La representación del valor -20.5 es:

1   100 0000 0011   0010 0100 0000 0000 …. 0000