Los números reales en coma flotante se convierten a binario en tres pasos:

1. Convertir al sistema binario
2. Escribir en notación científica
3. Seguir el standard IEEE754 para 32 bits

Por una parte la parte entera del número real se convierte a binario y por otra la parte fraccionaria, según el algoritmo que se explica en el vídeo https://www.youtube.com/watch?v=VMcypTxcbvY. Este algoritmo deberá ser el utilizado, **no permitiéndose** el uso de otros algoritmos.

En esta práctica, apoyándonos en la **anterior**, y siguiendo las especificaciones de la **Figura 1**, se realizarán los siguientes pasos:

1. Convertir un número real con signo a **binario**
2. Escribir el número binario en **notación científica**
3. Representar el número binario según el **standard IEEE754** para 32 bits

```
numero real en base decimal ? -134.3125
numero real convertido a binario:
  10000110.0101
numero binario en notacion cientifica:
  1.00001100101
exponente: 7
exponente+127 en binario:
  10000110
representacion en memoria:
  11000011000001100101000000000000
```

Figura 1. Ejemplo de ejecución del programa

Se deben usar los prototipos, y defines, indicados en el siguiente recuadro:

```c
                        // ESPERANZA MACARENA PLAZA MARTINEZ
                        // Nº de matricula: br0427


#include <stdio.h>
#include <stdlib.h>
#define maximoChars 64
#define N 32

//Prototipos
//parctica 6

void binarioEntera(int, char []);
// un int se convierte a binario (se almacena en el array)
```

```
void binarioFraccionaria(float , char []);
// un float se convierte a binario (se almacena en el array)

void resetear(char []);
// se resetea el array

void ponerPosicion (char [],int , char );
// se coloca un char en la posicion int del array

int bitsBlanco(char []);
// chars en blanco en el array

void insertarFinal(char [],char );
// se inserta un char al final del array, desplazando el resto a la izquierda

void printBinario(char []);
// se printa el array con los char del numero binario

//practica 7

void moverIzda(char []);
// mueve a la izda todos los bits del array

void scanReal(float * );
// scan del número real a convertir

int posicionPuntoDecimal (char []);
// posicion en el array de '.'

void notacionCientifica(char [], int * );
// convierte el binario en notacion cientifica, transmitiendo el exponente

void copiarMantisa(char [],char []);
// copia la mantisa a un array de 32 chars, en las últimos 23 posiciones del
array

void copiarExponente(char [],char []);
// copia el exponente a un array de 32 chars, en las 8 siguientes posiciones a
la posicion 0

void colocarSigno(char ,char []);
// coloca el signo en un array de 32 chars, en la posicion 0
```

```c
int main() {
    int exponente;
    float decimal;
    char binary[maximoChars], IEEE754[32];

    scanReal(&decimal);
    int parteEntera = (int) decimal;
    decimal = decimal - (int) decimal;
    resetear(binary);
    binarioEntera(parteEntera, binary);
    binarioFraccionaria(decimal, binary);
    printf("\nNumero: %.3f convertido a binario:\n", decimal);
    printBinario(binary);
    //Standard IEEE754
    resetear(IEEE754);
    if(decimal < 0){
        colocarSigno('1', IEEE754);
    } else{
        colocarSigno('0', IEEE754);
    }
    printf("\nNumero: %.3f convertido en notacion cientifica:\n", decimal);
    notacionCientifica(binary, &exponente);
    printBinario(binary);
    printf("\nExponente: %d", exponente);
    printf("\nExponente +127 en binario:");
    binarioEntera(exponente + 127, IEEE754);
    printBinario(IEEE754);
    copiarExponente(binary, IEEE754);
    copiarMantisa(binary, IEEE754);
    printf("\nRepresentacion en memoria:\n");
    for(int i = 0; i < 32; i++){
        printf("%c", IEEE754[i]);
    }

    return 0;
}

//Definicion de funciones

void binarioEntera(int num, char binary[]){
    int i = maximoChars - 1;
```

```
    while(num > 0){
        if(10 * (((float) num / 2) - (num / 2)) >= 5){
            binary[i--] = '1';
        } else{
            binary[i--] = '0';
        }
        num /= 2;
    }
}


void binarioFraccionaria(float decimal, char binary[]){
    insertarFinal(binary, '.');
    while(decimal != 0.0){
        if((int) (decimal * 2) != 0){
            insertarFinal(binary, '1');
        } else{
            insertarFinal(binary, '0');
        }
        decimal = decimal * 2 - ((int) (decimal * 2));
    }
}


void resetear(char binary[]){
    for(int i = 0; i < maximoChars; i++){
        binary[i] = ' ';
    }
}


void ponerPosicion (char binary[],int pos, char car){
    binary[pos] = car;
}


int bitsBlanco(char cad[]){
    int count = 0;

    while(count < maximoChars && cad[count] == ' '){
        count++;
    }
    return count;
}
```

```c
void insertarFinal(char cad[], char car){
    for (int i = 0; i < maximoChars - 1; i++) {
        cad[i] = cad[i + 1];
    }
    cad[maximoChars - 1] = car;
}

void printBinario(char binary[]){
    for(int i = 0; i <= maximoChars; i++){
        if(binary[i] != ' '){
            printf("%c", binary[i]);
        }
    }
}

void moverIzda(char cadena[]){
    for(int i = bitsBlanco(cadena) - 1; i < maximoChars - 1; i++){
        cadena[i] = cadena[i + 1];
    }
}

void scanReal(float *decimal){
    printf("\nEscriba la parte decimal de un numero: ");
    fflush(stdin);
    scanf("%f", decimal);
}

int posicionPuntoDecimal (char cadena[]){
    int posicion = 0;
    int i = bitsBlanco(cadena)

    while(i < maximoChars && posicion != 0){
        if(cadena[i] == '.'){
            posicion = i;
        } else{
            i++;
        }
    }

    return posicion;
}
```

```
void notacionCientifica(char cadena[], int * exponente){
    int count = 0;

    for(int i = posicionPuntoDecimal(cadena); i > bitsBlanco(cadena); i--){
        cadena[i] = cadena[i - 1];
        count++;
    }
    cadena[bitsBlanco(cadena) + 1] = '.';
    *exponente = count - 1;
}

void copiarMantisa(char cadena[],char mantisa[N]){
    int i = 9;
    for(int j = posicionPuntoDecimal(cadena) + 1; i < maximoChars; i++, j++){
        mantisa[i] = cadena[j];
    }

    for(int i = 0; i < 32; i++){
        if(mantisa[i] == ' ' || mantisa[i] != '0' && mantisa[i] != '1' &&
mantisa[i] != ' '){
            mantisa[i] = '0';
        }
    }
}

void copiarExponente(char cadena[], char exponente[N]){
    int i = 1;
    int exponent = 0;

    for(exponent = bitsBlanco(cadena); i < 10; i++, exponent++){
        exponente[i] = cadena[exponent];
    }
}

void colocarSigno(char signo, char cadena[N]){
    cadena[0] = signo;
}
```