

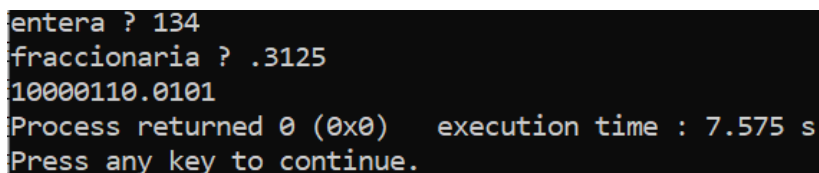
Los números reales en coma flotante se convierten a binario en tres pasos:

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 se hará lo mismo que en la práctica 3, pero definiendo y usando funciones y arrays. La conversión a binario se hará usando un array con un máximo de chars definido en la parte #define del programa.



```
entera ? 134
fraccionaria ? .3125
10000110.0101
Process returned 0 (0x0) execution time : 7.575 s
Press any key to continue.
```

Figura 1. Ejemplo de ejecución del programa

Se deben usar los prototipos indicados en el siguiente recuadro.

```
// defines
#define maximo_chars 64

// prototipos
void binario_entera(int, char [maximo_chars]);
// un int se convierte a binario (se almacena en el array)

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

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

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

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

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

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

int main(){
    char array[maximo_chars];
    int numEnt,j,x=0;
    char c, punto = '.';
    float numDec;
    do{
        printf("entero [0-255]?:" );
        scanf("%i", &numEnt);
    }while(numEnt<0 || numEnt>255);

    do{
        printf("\ndecimal?:" );
        scanf("%f", &numDec);
    }while(numDec<0 || numDec>=1);
    printf("\n\n");

    resetear(array);
    binario_entera(numEnt,array);
    insertar_final(array,punto);
    bits_blanco(array);
    x = (maximo_chars-bits_blanco(array));

    for(j=0; j<=(maximo_chars-bits_blanco(array)); j++){
        c = array[maximo_chars-j];
        poner_posicion(array, x, c);
        x--;
        array[maximo_chars-j] = ' ';
    }

    binario_fraccionaria(numDec, array);
    prn_binario(array);
    return 0;
}

// definicion de las funciones
void binario_entera(int numEnt, char array[maximo_chars]){
    int i = maximo_chars-1;
    float floatnumEnt;
    floatnumEnt=(int)numEnt;
    while((int)floatnumEnt){
        floatnumEnt/=2;
        array[i]=(((floatnumEnt-(int)floatnumEnt))>=0.5)?'1':'0';
        i--;
    }
}

void binario_fraccionaria(float numDec, char array[maximo_chars]){
```

```
int i = maximo_chars;
while(array[i]!=' '){
    i--;
}
while ((numDec - (int)numDec) != 0){
    numDec *= 2;
    array[i] = (((int)numDec) % 2) == 0 ? '0':'1';
    numDec = numDec - (int)numDec;
    i++;
}
}

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

void poner_posicion (char array[maximo_chars],int pos, char c){
    array[pos]=c;
}

int bits_blanco(char array[maximo_chars]){
    int i,contador=0;
    for(i=0;i<maximo_chars;i++){
        if(array[i] == ' ')
            contador++;
    }
    return(contador);
}

void insertar_final(char array[maximo_chars],char c){
    int i;
    for(i=0;i<maximo_chars;i++){
        array[i] = array[i+1];
        array[maximo_chars-1] = c;
    }
}

void prn_binario(char array[maximo_chars]){
    int i;
    for(i=0;i<maximo_chars;i++){
        printf("%c",array[i]);
    }
}
```