Au sujet d'une conversation sur les nombres réels écrits en Simple Précision (SP) ou en Double Précision (DP) dans un programme, voici une petite démonstration des résultats que cela peut donner lorsque l'on ne fait pas attention à ce que l'on croit n'être que maniaqueries de numériciens ! (cette note de cours améliore donc grandement mon exemple de cours au DEA d'Astrophysique).

On se place dans la représentation IEEE des nombres flottants du type '*big endian*' c'est-à-dire d'abord le bit de signe (souligné par le caractère '|'), puis l'exposant (biaisé) (dont les bits sont soulignés par les caractères '^^^^^^') puis enfin la mantisse (dans son ordre naturel).

L'exemple est construit autour des valeurs *machine* du nombre décimal `0.1` qui, il faut le remarquer, n'est pas un nombre représentable exactement en binaire comme on va le voir dans la partie suivante.

D'abord des résultats mathématiquement exacts :

$$0.1 = 9\sum_{i=1}^{\infty}\left(\frac{1}{16}\right)^{i} - 0.5$$

d'où la représentation binaire (infinie) de `0.1` :

```
0.1_10 = 0.0001 1001 1001 1001 1001 .... _2
       = 0.19999 .... _16
```

Avec les arrondis IEEE de la SP, `0.1` (Fortran) ou `0.1f` (C) s'écrit donc :

En binaire :
```
00111101111001100110011001101
|^^^^^^^^
```

ce qui représente la valeur exacte de :

$$0.1(SP) = 0.1 + \frac{2}{5}\left(\frac{1}{16}\right)^{7}$$

Avec les arrondis IEEE de la DP, `0.1d0` (Fortran) ou `0.1` (C) s'écrit donc :

En binaire :
```
0011111110111001100110011001100110011001100110011001100110011010
|^^^^^^^^^^^
```

ce qui représente la valeur exacte de :

$$0.1(DP) = 0.1 + \frac{2}{5}\left(\frac{1}{16}\right)^{14}$$

Si on aligne les mantisses des 2 nombres :

```
SP : 10011001100110011001101
DP : 10011001100110011001100110011001100110011001100110011010
```

on voit bien que le `0.1(SP)` est un tout petit plus grand que `0.1 (DP)`

On peut aussi comparer l'erreur de représentation et le *spacing* :

L'erreur de représentation est définie par `0.1 - 0.1(P)` où `P` est la précision.

	Erreur de représentation	*Spacing* (au voisinage de 0.1)
SP	≈ 1.490 E-09	≈ 7.451 E-09
DP	≈ 5.551 E-18	≈ 13.88 E-18

A titre d'exemple, voici un programme Fortran mettant en oeuvre ce qui vient d'être dit :

```
Program ZERO_ONE
   Double Precision :: x
   !
   Write(*,*) " 0.1 en SP et en binaire : "
   Write(*,"(B32)" ) 0.1
   Write(*,"(B32)" ) REAL( 0.1d0 - 0.4d0 * 0.0625d0**7 )      ! verification
   Write(*,*)
   !
   Write(*,*) " 0.1 en DP et en binaire : "
   Write(*,"(B64)" ) 0.1d0
   Write(*,"(B64)" ) 0.1d0 - 0.4d0 * 0.0625d0**14        ! verification ===>
   Write(*,*)                                            ! cancellation
   !                                                     !    des bits !!
   Write(*,*) " On se sert de la fonction Spacing pour des comparaisons : "
   Write(*,*) SPACING(0.1), SPACING(0.1d0)
   Write(*,*) 0.4 * 0.0625**7 , 0.4d0 * 0.0625d0**14
   Write(*,*)
   !
   Write(*,*) " Pour aller jusqu'au dernier bit : on peut le faire comme ce"
   Write(*,*)"     qui va suivre car on travaille bien en base 2 "
   x = 2.5d0 * ( 0.1 - 0.1d0 ) * 16.d0**7            ! valeur theorique de x
   Write(*,*) x                                      !   x = 1 - (1/16)^7
   Write(*,*) 1.d0 - 0.0625d0**7                     !   ====> OK
   !
   Write(*,"(A)",Advance="No") "                      et encore mieux : "
   Write(*,*) REAL( ( 1.d0 - x ) * 16.d0**7 )
   Write(*,*)
   !
   STOP " PARFAIT !! "
End Program ZERO_ONE
```

et son listing de sortie :

```
0.1 en SP et en binaire :
  11111011001100110011001101
  11111011001100110011001101

0.1 en DP et en binaire :
 1111111011100110011001100110011001100110011001100110011001100110011010
 1111111011100110011001100110011001100110011001100110011001100110011010

On se sert de la fonction Spacing pour des comparaisons :
  7.4505806E-09  1.387778780781446E-017
  1.4901161E-09  5.551115123125783E-018

Pour aller jusqu'au dernier bit : on peut le faire comme ce
    qui va suivre car on travaille bien en base 2
0.999999996274710
0.999999996274710
                    et encore mieux :    1.000000

 PARFAIT !!
```