LES PROBLÈMES DE PORTABILITÉ DU C

Il existe 2 problèmes principaux à la portabilité des applications écrites en C, cela dû en partie à une insuffisance de contraintes dans la Norme (raisons historiques principalement).

1) Le type“ *int* ”:

Rien ne dit quelle est la taille (en unité de *char* , retournée par la fonction *sizeof*) correspondant à une variable de ce type. Dans le tableau suivant, on y indique cette taille pour différents systèmes (le mot « système » représente ici le couple « ordinateur et compilateur »). Dans ce même tableau, on y indique pour des comparaisons, la taille des “ *short int* ” et des “ *long int* ”. On y a fait aussi figurer la taille des données de type “ *size_t* ” , “ *clock_t* ” et “ *time_t* ”. À remarquer que “ *int* ” n’est pas nécessairement synonyme de “ *short int* ” ou de “ *long int* ” .

	short int	*int*	*long int*	*size_t*	*clock_t*	*time_t*	type de *char*
PC CVI	2	4	4	4	4	4	signed
PC Borland	2	2	4	2	4	4	signed
SGI *mode 32bits*	2	4	4	4	4	4	unsigned
SGI *mode 64 bits*	2	4	8	4	4	4	unsigned
Dec-Alpha	2	4	8	8	4	4	signed

2) Le caractère signé ou non-signé du type“ *char* ”:

Rien ne dit si le type “ *char* ” est considéré, par défaut, par le système, comme une valeur signée ou non-signée. On voit dans le tableau quel sont les choix par défaut des mêmes systèmes que précédemment.

Remarque : **Par définition, *sizeof*(*char*) = 1** ; ce qui ne veut pas forcement dire que *char* est toujours représenté par un seul octet !