Introduction au C

Part 1

Tout le monde vous a dit : « Laisse tomber le Pascal, fait du C c'est mieux » et naturellement vous vous êtes dit que vous allez vous y mettre.

Une des grandes différences entre le Pascal et le C c'est que vous pouvez tout faire, et c'est là le problème ! Si vous n'avez pas un minimum d'organisation, votre programme fonctionnera peut-être mais bonjour les résultats et je ne vous parle pas de la maintenance…

Pour toutes suggestions/corrections : baygon@alrj.org

Le C est CASE SENSITIVE (Il fait la différence entre les majuscules et les minuscules)

Commençons doucement.

```
#include <stdlib.h>
#include <stdio.h>

int main(){
	printf("Hello World\n");
}
```

Pour compiler : gcc –o nom_exécutable nom_source.c

KESSEKSSA?

Les *#include* sont au C ce que sont les *uses* au Pascal. Sauf que en C, la partie interface se trouve dans le fichier .h et l'implémentation dans le .c.

Les <> signifie que le fichier se trouve dans le répertoire par défaut si le fichier se trouve dans le même répertoire que le programme, alors on mettra ""

Miracle de la technologie, nous n'incluons que les en-têtes (prototypes) des fonctions.

main() est le point d'entrée du programme (c'est là que commencera l'exécution). De plus main est une fonction qui (dans ce cas-ci) ne prend pas de paramètre et qui renvoie un *int*.

Les {} sont les *Begin* et les *End* du Pascal.

printf : est une fonction qui affiche une chaîne de caractères. Le *\n* signifie que l'on désire passer à la ligne et *vider les buffers*. En C, les opérations d'écritures sont bufférisée et le buffer n'est vidé que quand il est plein, pas cool car *Hello World* ne remplit certainement pas un buffer de 1024 caractères donc on le vide.

Un peu plus loin

Maintenant, nous allons lire une chaîne de caractères au clavier et l'afficher.
En C les commentaires commencent par /* et finissent par */

```
#include <stdio.h>
#include <stdlib.h>

int main(){
	char phrase[20]; /*On déclare une chaîne de 20 caractères*/

	printf("Quel est votre prénom ? \n");
	gets(phrase);/*On lit sur l'entrée standard*/
	printf("Bonjour %s\n",phrase);
}
```

CEKOICA ?

Nous déclarons une variable *phrase* qui est en fait un tableau de *char* ➔ une chaîne de caractère. En C, il n'y a pas de différences entre une chaîne et un tableau.

Notre variable *char* est locale à la fonction *main()* pour une variable globale, nous l'aurions déclarée au-dessus !

gets lit une chaîne de caractère sur l'entrée standard (en général le clavier).

printf affiche la variable *phrase*. le *%s* lui dit qu'on veut afficher une *string*.

Ce qui m'amène à vous parler des types de base en C

Les entiers
char
short
int
long
unsigned short
unsigned int ⇔ unsigned
unsigned long

Les réels
float
double
long double (ANSI C)

Un dernier exemple

```
#include <stdlib.h>
#include <stdio.h>

int main(){
      int a;
      char nombre[5];

      printf("Entrez un nombre : \n");
      gets(nombre);
      a = atoi(nombre);
      printf("\"%i * 2 = %i\"",a,a*2);
}
```

atoi : convertit une chaîne ASCII en Int.

- itoa
- atol
- ltoa (suis pas sûr)

le \" indique qu'on veut afficher un ".

Ce qui m'amène à vous parler des formats d'affichage du *printf*.

Formats	
`%d ou %i`	Entier en décimal
`%o`	Entier en octal
`%x`	Entier en hexa
`%X`	Entier en hexa (avec les lettres en MAJ)
`%f`	2.5
`%g`	le plus court des deux (%f, %e)
`%e`	2.5e+0
`%c`	1 caractère
`%s`	Une chaîne de caractère

Dernier point pour cette fois

```
#include <stdio.h>
#include <stdlib.h>

int main(){
      char car;

      car = 'A'
      printf(%i\n",car);
}
```

Que va afficher le programme ?

.
.
.
.

65

Ben oui, si j'avais voulu afficher *A* j'aurais mis *`%c`* et non *`%i`* .