1 Generalites

```
Syntaxe:        #include <stdarg.h>
                void va_start(va_list ap, param);
                type va_arg(va_list ap, type);
                void va_end(va_list ap);
```

Description :

- la macro `va_start` fait pointer `ap` sur le premier argument variable fourni à la fonction. param étant le nom du dernier paramètre nommé.
- la macro `va_arg` renvoie le premier argument variable et fait pointer `ap` sur l'argument suivant. `type` est le type de l'argument qui va être lu et `va_arg` génère une expression de ce même type.
- la macro `va_end` remet tout en état normal avant le retour à la fonction appelante.

Remarque : le problème à résoudre consiste à définir une méthode de calcul du nombre d'arguments réels de la fonction appelante, de manière à pouvoir déterminer le nombre d'arguments à traiter:

- Le premier argument peut être le nombre de paramètres réels.
- un argument devra avoir une valeur particulière (0 par exemple) afin d'indiquer la fin des arguments variables (c.f. exemple 1).
- Le premier argument doit être structuré d'une telle façon que la fonction appelée doit pouvoir compter les arguments variables (c.f. exemple 2).

2 Exemple 1

```
/* calcule la somme de n entiers     */
/* le dernier argument doit être à 0 */
#include <stdio.h>
#include <stdarg.h>
#include <stdlib.h>

int somme(int n1, ...) {
  va_list  pa;
  int som, n;
  som = n1;
  va_start(pa, n1);
  while( (n = va_arg(pa, int)) != 0)
     som = som + n;
  va_end(pa);
  return som;
}

int main() {
  printf("1 + 3 + 5 + 7 + 9 = %d\n", somme(1,3,5,7,9,0));
  printf("1 + 1 = %d\n", somme(1,1,0));
  return EXIT_SUCCESS;
}

/*-- résultat de l'exécution ---------------------------
1 + 3 + 5 + 7 + 9 = 25
1 + 1 = 2
------------------------------------------------------*/
```

3 Exemple 2

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

void myprintf(char *format, ...) {
   va_list pa;
   int n;
   char *s, c;
   float f;

   va_start(pa, format);
   while (*format != '\0') {
     if ( *format == '%' ) {
       switch (*++format) {
         case '%' : putchar('%'); break;
         case 'c' : /* affichage d'un caractère */
                    c = va_arg(pa, char);
                    putchar(c);
                    break;
         case 'd' : /* affichage d'un entier */
                    n = va_arg(pa, int);
                    printf("%d", n);
                    break;
         case 'f' : /* affichage d'un float */
                    f = va_arg(pa, double);    /* !!!!! */
                    printf("%f", f);
                    break;
         case 's' : /* affichage d'une chaîne */
                    s = va_arg(pa, char *);
                    for ( ; *s != '\0'; s++ )
                      putchar( *s );
                    break;
       } /* end switch */
     }else
       putchar( *format );
     format++;
   }
   va_end(pa);
}

int main() {
   myprintf("float = %f\n", (float) 1.2345);
   myprintf("int = %d   char = %c   Chaine = %s\n",
                 123, 'A', "C is beautiful !" );
   return EXIT_SUCCESS;
}

/*-- résultat de l'exécution ---------------------------------
float = 1.234500
int = 123   char = A   Chaine = C is beautiful !
------------------------------------------------------------*/
```