Nom	Notes
`void *memcpy(void *dest, const void *src, size_t n);`	copie *n* octets entre deux zones mémoire, qui ne doivent pas se superposer
`void *memmove(void *dest, const void *src, size_t n);`	copie *n* octets entre deux zones mémoire ; à la différence de memcpy, les zones mémoire peuvent se superposer
`void *memchr(const void *s, int c, size_t n);`	retourne en pointeur la première occurrence *c* parmi les *n* premiers octets de *s*, ou *NULL* si *c* n'est pas trouvé
`int memcmp(const void *s1, const void *s2, size_t n);`	compare les *n* premiers caractères de deux zones mémoire
`void *memset(void *, int, size_t);`	remplit une zone mémoire de la répétition d'un caractère
`char *strcat(char *dest, const char *src);`	concatène la chaîne `src` à la suite de dest
`char *strncat(char * dest, const char * src, size_t n);`	concatène au plus n caractères de la chaîne `src` à la suite de `dest`
`char *strchr(const char *, int);`	cherche un caractère dans une chaîne et renvoie un pointeur sur le caractère, en cherchant depuis le début
`char *strrchr(const char *, int);`	idem que strchr, recherche à partir de la fin
`int strcmp(const char *, const char *);`	compare deux chaînes lexicalement
`int strncmp(const char *, const char *, size_t n);`	compare les *n* premiers octets au plus de deux chaînes en utilisant l'ordre lexicographique
`int strcoll(const char *, const char *);`	compare deux chaînes en utilisant l'ordre lexicographique
`char *strcpy(char *toHere, const char *fromHere);`	copie une chaîne de caractères d'une zone à une autre
`char *strncpy(char *toHere, const char *fromHere, size_t n);`	copie au plus *n* caractères d'une chaîne d'une zone à une autre
`char *strerror(int);`	retourne la chaîne de caractères correspondant à un numéro d'erreur
`size_t strlen(const char *);`	retourne la longueur d'une chaîne caractères
`size_t strspn(const char *s, const char *accept);`	détermine la taille de la sous-chaîne initiale maximale de *s* ne contenant que des caractères présents dans `accept`
`size_t strcspn(const char *s, const char *reject);`	détermine la taille de la sous-chaîne initiale maximale de *s* ne contenant pas de caractères de `reject`
`char *strpbrk(const char *s, const char *accept);`	trouve la première occurrence d'un caractère d'`accept` dans *s*
`char *strstr(const char *haystack, const char *needle);`	trouve la première occurrence de la chaîne `needle` dans la chaîne `haystack`
`char *strtok(char *, const char *);`	scinde une chaîne en éléments lexicaux. Note: la fonction modifie la chaîne passée en paramètre.
`size_t strxfrm(char *dest, const char *src, size_t n);`	transforme `src` de façon que le tri par ordre lexicographique de la chaîne transformée soit équivalent au tri par ordre lexicographique de `src`.