

```

1  #include <stdio.h>
2  void swap1(int, int);
3  void swap2(int *, int *);
4  void swap1(int a, int b) {
5      int c;
6      c = a;
7      a = b;
8      b = c;
9  }
10 void swap2(int * a, int * b) {
11     int c;
12     c = *a;
13     *a = *b;
14     *b = c;
15 }
16 int main(void) {
17     int a, b;
18     a = 1; b = 2;
19     printf("a = %d, b = %d\n", a, b);
20     printf("appel de la fonction swap1(a, b)\n");
21     swap1(a, b);
22     printf("a = %d, b = %d\n", a, b);
23     printf("*****\n");
24     a = 1; b = 2;
25     printf("a = %d, b = %d\n", a, b);
26     printf("appel de la fonction swap2(&a, &b)\n");
27     swap2(&a, &b);
28     printf("a = %d, b = %d\n", a, b);
29     return 0;
30 }

```

```

1  -bash-2.05b$ gcc -Wall swap01.c -o swap01
2  -bash-2.05b$ ./swap01
3  a = 1, b = 2
4  appel de la fonction swap1(a, b)
5  a = 1, b = 2
6  *****
7  a = 1, b = 2
8  appel de la fonction swap2(&a, &b)
9  a = 2, b = 1

```

FIG. 1 – Pointeurs et paramètres de fonction.

```

1  void swap(int * v1, int * v2) {
2      *v1 ^= *v2;
3      *v2 ^= *v1;
4      *v1 ^= *v2;
5  }

```

FIG. 2 – Version améliorée de la fonction d'échange de variables.

```

1  #include <stdio.h>
2
3  void imprimer(char * s);
4
5  void imprimer(char * s) {
6      while(*s)
7          putchar(*s++);
8  }
9
10 int main(void) {
11     char chaine[] = {'H', 'e', 'l', 'l', 'o', ' ', 'W', 'o', 'r', 'l', 'd', ' ', '!', '\n', '\0'};
12     imprimer(chaine);
13     return 0;
14 }

```

```

1  #include <stdio.h> /* Pour putchar */
2  #include <stdlib.h> /* Pour malloc et free */
3  #include <string.h> /* Pour strcpy */
4
5  void imprimer(char * s);
6  char * allouer_chaine(int n);
7
8  int main(void) {
9      char * str = NULL;
10     str = allouer_chaine(20);
11     strcpy(str, "Hello World !\n");
12     imprimer(str);
13     free(str);
14     return 0;
15 }
16
17 char * allouer_chaine(int n) {
18     char * s;
19     s = malloc(n * sizeof * s);
20     if(s == NULL) {
21         fprintf(stderr, "Erreur d'allocation memoire.\n");
22         exit(1);
23     }
24     return s;
25 }
26
27 void imprimer(char * s) {
28     while(*s)
29         putchar(*s++);
30 }

```

FIG. 3 – Pointeurs et chaines de caractères.

Devoir 04 :

Écrire une fonction qui prend en argument une chaîne de caractères, alloue (`malloc`) une zone mémoire suffisante et enfin écrit la chaîne inversée dedans. La fonction retournera le pointeur vers la nouvelle chaîne créée.

Écrire la fonction `main` appelant cette fonction en lui donnant comme argument l'entrée clavier de l'utilisateur. Imprimer la chaîne retournée par la fonction et enfin libérer l'espace mémoire alloué.

Ce `main` permet d'inverser plusieurs chaînes dans la même exécution (une boucle). Le programme s'arrêtera quand l'utilisateur entrera le mot `quit`.

(lire les man de : `strcpy`, `strncpy`, `memcpy`, `strdup`, `strcmp`, `strncmp`, `malloc`, `free`.)