TD 2 - Les Pointeurs

Exercice 1

Ecrire un programme dans lequel vous :

1. Déclarez un entier *i* et un pointeur vers un entier *p*,
2. Initialisez l'entier à une valeur arbitraire et faites pointer le pointeur sur *i*,
3. Affichez la valeur de *i*,
4. Modifierez l'entier pointé par *p* (en utilisant *p* et non pas *i*),
5. Et imprimez la valeur de *i* une dernière fois.

Solution

```
#include <stdio.h>

int main()
{
  /* Déclaration d'un entier pointé */
  int i;
  /* Déclaration d'un pointeur sur un entier */
  int* p_i;

  /* Initialisation de i */
  printf("Saisir un nombre : ");
  if(scanf("%d",&i) != 1) {
       printf("ce n'est pas un entier");
       return 1;
  }

  /* iPointeur pointe vers i */
  p_i = &i;

  /* Affichage de la valeur de i */
  printf("valeur de i avant : %d\n",i);

  /* Changement de la valeur du contenu pointé par i */
  (*p_i) += 2;

  /* Affichage de la valeur de i */
  printf("valeur de i après: %d\n",i);
  return 0;
}
```

Amélioration : Modifier la solution pour que la saisie incorrecte ne fait pas sortir du programme mais redemande de rentre un nombre.

Exercice 2 :

On vous donne le programme suivant :

```
int main()
   {
      int A = 1;
      int B = 2;
      int C = 3;
      int *P1, *P2;
      P1 = &A;
      P2 = &C;
      *P1 = (*P2)++;
      P1 = P2;
      P2 = &B;
      *P1 -= *P2;
      ++*P2;
      *P1 *= *P2;
      A = ++*P2**P1;
      P1 = &A;
      *P2 = *P1 /= *P2;
      return 0;
   }
```

Complétez le tableau suivant pour chaque instruction du programme ci-dessous.

	A	B	C	P1	*P1	P2	*P2
Init.	1	2	3	/	/	/	/
P1 = &A							
P2 = &C							
*P1 = (*P2)++							
P1 = P2							
P2 = &B							
*P1 -= *P2							
++*P2							
*P1 *= *P2							
A = ++*P2 * *P1							
P1 = &A							
*P2 = *P1 /= *P2							

Solution

	A	B	C	P1	*P1	P2	*P2
Init.	1	2	3	/	/	/	/
P1 = &A	1	2	3	&A	1	/	/
P2 = &C	1	2	3	&A	1	&C	3
*P1 = (*P2)++	3	2	4	&A	3	&C	4
P1 = P2	3	2	4	&C	4	&C	4
P2 = &B	3	2	4	&C	4	&B	2
*P1 -= *P2	3	2	2	&C	2	&B	2
++*P2	3	3	2	&C	2	&B	3
*P1 *= *P2	3	3	6	&C	6	&B	3
A = ++*P2 * *P1	24	4	6	&C	6	&B	4
P1 = &A	24	4	6	&A	24	&B	4
*P2 = *P1 /= *P2	6	6	6	&A	6	&B	6

Attention :

***P1 = (*P2)++ ; signifie que *P2 est recopié dans *P1 et ensuite que *P2 est incrémenté.**

A = ++*P2 * *P1 ; signifie par contre qu'on commence à incrémenter *P2 puis ensuite que (*P2) * (*P1) est calculé et le résultat stocké dans A.

***P2 = *P1 /= *P2 ; signifie qu'on fait d'abord *P1 = *P1 / *P2 puis qu'on recopie le résultat dans *P2.**

Exercice 3 :

Ecrire un programme qui permet d'échanger deux valeurs réelles :

1. Ecrire une fonction ***void echanger (float *, float *)*** qui échange les 2 valeurs,
2. Ecrire la fonction ***int main()*** qui demande à l'utilisateur de rentrer 2 valeurs réelles (en virgule flottante), qui les affiche, puis appelle la fonction ***echanger*** avant d'afficher à nouveau les 2 valeurs.

Solution

```
#include <stdio.h>

/* prototype de la fonction echanger */
void echanger(float *, float *);

int main() {
  /* les 2 réels */
  float x, y;

  printf("Donnez deux réels : \n");
  /* saisie des réels */
  if (scanf("%f %f", &x, &y) != 2) {
    printf("Ce ne sont pas des réels");
    return 1;
  }

  /* affichage avant échange */
  printf("Avant échange, dans x : %f ; dans y : %f\n", x, y);
  /* appel de la fonction qui échange le contenu des variables */
  echanger(&x, &y);
  /* affichage après échange */
  printf("Après échange, dans x : %f ; dans y : %f\n", x, y);

  return 0;
}

/* définition de la fonction */
void echanger(float *ad_f1, float *ad_f2) {
  /* variable qui sert à sauvegarder une des deux valeurs */
  float tampon;
  /* on sauvegarde la valeur de la variable pointée par ad_f1 */
  tampon = *ad_f1;
  /* on recopie la valeur de la variable pointée par ad_f2
  dans la variable pointée par ad_f1 */
  *ad_f1 = *ad_f2;
  /* on recopie la valeur sauvegardée dans la variable pointée par ad_f2 */
  *ad_f2 = tampon;
}
```

Amélioration : Modifier la solution pour que la saisie incorrecte ne fait pas sortir du programme mais redemande de rentre un nombre.

Exercice 4 :

Ecrire un programme qui vérifie si la chaine de caractères *CH* entrée par l'utilisateur est un palindrome ou non.
Un palindrome est un mot ou groupe de mots qui peut être lu indifféremment de gauche à droite ou de droite à gauche. Par exemple PIERRE n'est pas un palindrome, OTTO est un palindrome.
Vous utiliserez la fonction ***fgets(CH, 20, stdin)*** (si CH est une chaîne de 20 caractères au maximum en comptant \0) afin de stocker la chaine saisie par l'utilisateur (plus rapide et sûre que *scanf*).

Solution

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

int main() {
  /* Déclarations */
  char CH[101]; /* chaîne donnée    */
  char *P1, *P2; /* pointeurs d'aide */
  int PALI; /* indicateur logique:          */
  /* vrai si CH est un palindrome */

  /* Saisie des données */
  printf("Entrez une ligne de texte (max.100 caractères) :\n");
  if(fgets(CH, 101, stdin) == NULL) {
    printf("Erreur lors de la lecture de la ligne de texte.");
    return 1;
  }
  /* Placer P2 sur la dernière lettre de la chaîne. */
  /* On commence par pointer au début et on parcourt la chaîne */
  /* La ligne lue par fgets se termine par \n ou par \0 */
  for (P2 = CH; (*P2 != '\n') && (*P2 != '\0'); P2++)
    ;
  /* P2 pointe sur le \n ou sur le \0 donc on décrémente de 1 */
  P2--;
  /* Contrôler si CH est un palindrome */
  /* on suppose que c'est un palindrome puis on vérifie */
  PALI = 1;
  /* P1 démarre sur la première lettre
     P2 sur la dernière lettre
     P1 augmente et P2 diminue
     jusqu'à ce qu'on découvre que ce n'est pas un palindrome
     ou que P1 dépasse P2
  */
  for (P1 = CH; PALI && P1 < P2; P1++, P2--) {
    /* si les caractères sont différents */
    if (*P1 != *P2) {
      /* ce n'est pas un palindrome */
      PALI = 0;
    }
  }
  /* Affichage du résultat */
  if (PALI)
    printf("La chaîne \"%s\" est un palindrome.\n", CH);
  else
    printf("La chaîne \"%s\" n'est pas un palindrome.\n", CH);
  return 0;
}
```

Améliorations : 1) Modifier la solution pour que l'affichage se fasse correctement.
2) Réaliser une fonction palindrome() qui prend en paramètre la chaîne de caractères et renvoie si c'est un palindrome ou non