2014/202015
SMI-S3
Travaux Dirigés de Langage C
Série N°3

Exercice 1
Ecrire un programme qui lit un fichier et compte le nombre d'occurences de a, de e, de i, de o et de u. On utilisera l'instruction de branchement multiple switch.

Exercice 2
Ecrire un programme qui, pour une valeur x de type double, calcule la valeur numérique en x d'un polynôme de degré n, $P(X) = a_n X^n + ... + a_1 X + a_0$. Les valeurs de n, des coefficients a_i et de x sont entrées au clavier. On utilisera l'algorithme de Horner, qui évite les exponentiations (on calcule d'abord $a_n x + a_{n-1}$, puis $a_n x^2 + a_{n-1} x + a_{n-2}$ ...).

Exercice 3
Ecrire un programme qui lit un fichier et l'imprime à l'écran, en remplaçant tous les chiffres par le symbole *. On rappelle que le caractère de fin de fichier est la constante EOF définie dans la librairie standard stdio.h

Exercice 4
Ecrire un programme qui lit la dimension N d'un tableau T du type int (dimension maximale: 50 composantes), remplit le tableau par des valeurs entrées au clavier et affiche le tableau.
Calculer et afficher ensuite la somme des éléments du tableau.

Exercice 5
Ecrire un programme qui lit la dimension N d'un tableau T du type int (dimension maximale: 50 composantes), remplit le tableau par des valeurs entrées au clavier et affiche le tableau.
Effacer ensuite toutes les occurrences de la valeur 0 dans le tableau T et tasser les éléments restants. Afficher le tableau résultant

2014/202015
SMI-S3
Corrigé des Travaux Dirigés de Langage C
Série N°3

Solution 1

Ecrire un programme qui lit un fichier et compte le nombre d'occurences de a, de e, de i, de o et de u. On utilisera l'instruction de branchement multiple switch.

```
/*** calcul du nombre d'occurences de a, e, i, o et u  ***/
#include <stdio.h>
main()
{
 char c;
 int nb_a = 0, nb_e = 0, nb_i = 0, nb_o = 0, nb_u = 0;

 while ((c = getchar()) != EOF)
  {
   switch (c)
    {
    case 'a':
      nb_a++;
      break;
    case 'e':
      nb_e++;
      break;
    case 'i':
      nb_i++;
      break;
    case 'o':
      nb_o++;
      break;
    case 'u':
      nb_u++;
      break;
    }
  }
 printf("Nombre de a = %d\n",nb_a);
 printf("Nombre de e = %d\n",nb_e);
 printf("Nombre de i = %d\n",nb_i);
 printf("Nombre de o = %d\n",nb_o);
 printf("Nombre de u = %d\n",nb_u);
}
```

Solution 2

Ecrire un programme qui, pour une valeur x de type double, calcule la valeur numérique en x d'un polynôme de degré n, $P(X) = a_n X^n + ... + a_1 X + a_0$. Les valeurs de n, des coefficients a_i et de x sont entrées au clavier. On utilisera l'algorithme de Horner, qui évite les exponentiations (on calcule d'abord $a_n x + a_{n-1}$, puis $a_n x^2+a_{n-1}x + a_{n-2}$ ...).

```
/*** evaluation d'un polynome avec l'algo de Horner ***/
#include <stdio.h>
main()
{
 int i, n;
 double a, x, res;
```

```
  printf("Entrez le degre du polynome : ");
  scanf("%d",&n);
  printf("Entrez la valeur de x : ");
  scanf("%lf",&x);

  for (i = n; i >= 0; i--)
    {
      printf("Entrez le coefficient d'indice %d : ",i);
      scanf("%lf",&a);
      if (i == n)
        res = a;
      else
        {
          res *= x;
          res += a;
        }
    }
  printf("\n La valeur en %f du polynome est %f \n",x,res);
}
```

Solution 3

Ecrire un programme qui lit un fichier et l'imprime à l'écran, en remplaçant tous les chiffres par le symbole *.

```
/*******************************************************/
/*** remplacement des chiffres d'un fichier par *    ***/
#include <stdio.h>
main()
{
  char c;
  while ((c = getchar()) != EOF)
    {
      if (c >= '0' && c <= '9')
        putchar('*');
      else
        putchar(c);
    }
}
```

Solution 4

```
#include <stdio.h>
main()
{
 /* Déclarations */
 int T[50]; /* tableau donné  */
 int N;     /* dimension      */
 int I;     /* indice courant */
 long SOM;  /* somme des éléments - type long à cause */
            /* de la grandeur prévisible du résultat. */

 /* Saisie des données */
 printf("Dimension du tableau (max.50) : ");
 scanf("%d", &N );
 for (I=0; I<N; I++)
    {
     printf("Elément %d : ", I);
     scanf("%d", &T[I]);
```

```
    }
 /* Affichage du tableau */
 printf("Tableau donné :\n");
 for (I=0; I<N; I++)
    printf("%d ", T[I]);
 printf("\n");
 /* Calcul de la somme */
 for (SOM=0, I=0; I<N; I++)
    SOM += T[I];
 /* Edition du résultat */
 printf("Somme de éléments : %ld\n", SOM);
 return 0;
}
```

Solution 5

```
#include <stdio.h>
main()
{
 /* Déclarations */
 int T[50]; /* tableau donné    */
 int N;     /* dimension        */
 int I,J;   /* indices courants */

 /* Saisie des données */
 printf("Dimension du tableau (max.50) : ");
 scanf("%d", &N );
 for (I=0; I<N; I++)
    {
    printf("Elément %d : ", I);
    scanf("%d", &T[I]);
    }
 /* Affichage du tableau */
 printf("Tableau donné : \n");
 for (I=0; I<N; I++)
    printf("%d ", T[I]);
 printf("\n");
 /* Effacer les zéros et comprimer :         */
 /* Copier tous les éléments de I vers J et */
 /* augmenter J pour les éléments non nuls. */
 for (I=0, J=0 ; I<N ; I++)
    {
     T[J] = T[I];
     if (T[I]==0)
              J++;
    }
 /* Nouvelle dimension du tableau ! */
 N = J;
 /* Edition des résultats */
 printf("Tableau résultat :\n");
 for (I=0; I<N; I++)
    printf("%d ", T[I]);
 printf("\n");
 return 0;
}
```