Examen – CORRECTION - semestre 1
Initiation à la programmation 2014/2015
Électronique et Physique appliquée

Tous documents
non électroniques autorisés
durée 1h30

Nom, prénom :

Spécialité :

- **Répondre sur le document.**
- Écrire vos programmes en langage C en respectant la syntaxe et l'indentation usuelle ;
- Soignez votre présentation ;
- Les questions sont relativement indépendantes ;
- Chaque question est notée sur 2 points.

INDICATIONS A LA FIN

1. Analyse de code (2pts)

Qu'affiche le programme ci-dessous ?

```
#include<stdio.h>

int main (void) {
   int a  ;
    int tab[3] ;
    int *p ;
    for (a=0 ; a<3 ; a++ )
        tab[a]=a+1 ;
    p = tab   ;
    *p = 8 ;
    for (a=0 ; a<3 ; a++ )
        printf ("tab[%d] = %d ; \n", a, tab[a] ) ;
    p = &a ;
    *p = 9 ;
    printf ("a = %d, *p= %d\n", a, *p) ;
}
```

```
tab[0] = 8 ;
tab[1] = 2 ;
tab[2] = 3 ;
a = 9, *p= 9
```

2. Palindrome

On souhaite écrire un programme qui teste si une chaîne de caractères est un palindrome. Un palindrome est une chaîne de caractère dont l'ordre des lettres reste le même qu'on le lise de gauche à droite ou de droite à gauche.
Par exemple, « laval » est un palindrome. Aussi, dans les chaînes comportant des espaces, ces derniers ne sont pas comptés. La chaîne « rions noir » est accepté comme un palindrome

Voici un exemple de main de validation

```
int main (void) {
        char chaine1[]="laval"  ;
        char chaine2[]="rions noir" ;
        char chaine3[]="toto" ;

        afficheSiPalindrome(chaine1) ;
        afficheSiPalindrome(chaine2) ;
        afficheSiPalindrome(chaine3) ;
```

Voici la trace d'exécution :

```
laval est un palindrome.
rions noir est un palindrome.
toto n'est pas un palindrome
```

1) **Écrire** la fonction *estUnPalindrome* qui prend en paramètre une chaîne de caractères et retourne 1 si la chaîne est un palindrome et 0 sinon. (2pts)

```
int estUnPalindrome(char ch[]) {
      int i=0, j ;
      j=strlen(ch)-1 ;
      for(;;) {
            if (i>j) break ;
            if (ch[i]==' ') { i++ ; continue ;}
            if (ch[j]==' ') { j-- ; continue ;}
            if (ch[i]!=ch[j]) return 0 ;
            j-- ; i++ ;
      }
      return 1 ;
}
```

2) **Écrire** la fonction *afficheSiPalindrome* qui prend en paramètre une chaîne de caractères et affiche la chaîne suivie de, « est un palindrome » si ch est un palindrome, ou suivie de « n'est pas un palindrome » sinon. (2pts)

```
void afficheSiPalindrome(char ch[]) {
      printf ("%s ",ch) ;
      if (estUnPalindrome1 (ch))
            printf ("est un palindrome.\n");
      else
            printf ("n'est pas un palindrome\n");
}
```

3. tableaux et structures

On désire réaliser une bibliothèque de fonctions permettant d'effectuer des opérations sur des vecteurs de longueur quelconque.
Pour cela on déclare une structure de données appelée Vecteur dont la déclaration est donnée dans le code exemple suivant :

```
typedef struct sVecteur {
         int len ;
         int *tab ;
} Vecteur ;

Vecteur newVecteur( int l)  {
         Vecteur v;
         v.len = l ;
         v.tab= (int *) calloc (l, sizeof(int)) ;
         return (v) ;
}

int main (void) {
         Vecteur a;
         a=newVecteur(10) ;
         afficherVecteur (a) ;
         return(0) ;
}
```

Et voici une trace d'exécution :

```
[ 0 0 0 0 0 0 0 0 0 0 ]
```

1) **Expliquer** le rôle de la fonction *newVecteur. (1,5pts)*

La dimension des vecteurs étant quelconque, il faut réserver de la place pour le tableau dont le pointeur est dans la structure. De plus calloc initialise à zéro les données du tableau.

2) **Écrire** la fonction *afficherVecteur*. (2pts)

```
void afficherVecteur (Vecteur v)  {
      int i ;
      printf("[ ") ;
      for (i=0 ; i< v.len ; i++)
            printf ("%d ",v.tab[i]) ;
      puts("]") ;
}
```

3) **Écrire** la fonction *normeVecteur* qui retourne un nombre flottant et prend en paramètre un Vecteur. La valeur retournée est la norme du vecteur, c'est à dire la racine carrée de la somme des carrés des composantes : $\sqrt{x_0^2+x_1^2+x_2^2...}$ (2 pts)

```
float norme (Vecteur a)  {
      int i, n=0 ;
      for (i=0 ; i< a.len ; i++)
            n+= a.tab[i]*a.tab[i] ;
      return (sqrt(n)) ;
}
```

4) **Écrire** la fonction *void aleaVecteur(Vecteur *v)* qui modifie le vecteur *v* en lui attribuant des composantes aléatoires. (1,5 pts)

```
void aleaVecteur(Vecteur *v)  {
      int i ;
      for (i=0 ; i< (*v).len ; i++)
            (*v).tab[i]= rand();
}
```

5) **Écrire** la fonction *sommeVecteur* qui retourne la somme de 2 vecteurs. La somme de 2 vecteurs V1 et V2 est un vecteur de même dimension dont la composante de rang N est la somme des 2 composantes de même rang des vecteurs V1 et V2. (1,5 pts)

```
Vecteur sommeVecteur (Vecteur a, Vecteur b)  {
      int i ;
      Vecteur v;
      v = newVecteur(a.len) ;
      for (i=0 ; i< a.len ; i++)
            v.tab[i] = a.tab[i]+ b.tab[i] ;
      return (v) ;
}
```

6) **Écrire** la fonction `void saveVecteur(char * nomFichier, Vecteur v)` qui sauvegarde le vecteur dans un fichier texte. Sur la première ligne se trouvera la longueur du vecteur et sur la deuxième, ses coordonnées séparées par un espace. (2 pts)

```
void saveVecteur(char * nomFichier, Vecteur v) {
      int i ;
      FILE *fp ;
      fp = fopen( nomFichier, "w" ) ;
    if (fp == NULL ) {
            printf("Pb ouverture %s\n", nomFichier) ;
            return ;
      }
      fprintf (fp, "%d\n", v.len) ;

      for (i=0 ; i< v.len ; i++ )
            fprintf (fp, "%d ", v.tab[i]) ;
      fclose (fp) ;
      return ;
}
```

7) **Écrire** la fonction `loadVecteur` qui charge le vecteur à partir d'un fichier texte ayant le même format qu'à la question 7. Vous êtes libre de choisir le prototype de la fonction (type des paramètres et valeur de retour de la fonction) qui conviendra. (2 pts)

```
Vecteur loadVecteur(char * nomFichier) {
	Vecteur v ;
	v.len = 0 ;
	int i ;
	FILE *fp ;
	fp = fopen( nomFichier, "r" ) ;
	if (fp == NULL ) {
		printf("Pb ouverture %s\n", nomFichier) ;
		return(v) ; // retourne un Vecteur vide
	}
	fscanf (fp, "%d", &v.len) ;
	v = newVecteur (v.len) ;
	for (i=0 ; i< v.len ; i++ )
		fscanf (fp, "%d", &(v.tab[i])) ;
	fclose (fp) ;
	return v ;
}
```

8) **Écrire** une fonction main() permettant de valider ces fonctions. (1,5 pts)

```
int main (void) {
      Vecteur a,b;

      srand(time(NULL));
      a=newVecteur(10) ;
      b=newVecteur(10) ;
      aleaVecteur(&a) ;
      aleaVecteur(&b) ;
      afficherVecteur (a) ;
      afficherVecteur (b) ;
      afficherVecteur(sommeVecteur(a,b)) ;
      printf ("norme de a = %f\n", norme(a));
      saveVecteur("fic.txt", a) ;
      b = loadVecteur("fic.txt") ;
      afficherVecteur (b) ;
      return(0) ;
}
```