Partie 1 (6 points) Idées de correction

On voudrait afficher la figure suivante :

```
************* Som
*     1     * 1
*    555    * 15
*   88888   * 40
*  2222222  * 14
* 999999999 * 81
*88888888888* 88
* 555555555 * 45
*  7777777  * 49
*   33333   * 15
*    333    * 9
*     4     * 4
*************
```

Les chiffres à l'intérieur du cadre sont des nombres aléatoires entre 0 et 9.

La dimension du carré est un nombre demandé à l'utilisateur, sachant qu'elle doit être comprise entre 3 et 21 et qu'elle doit être impaire.

Les nombres sous Som sont la somme des chiffres de la ligne.

Ecrire un programme qui demande à l'utilisateur de saisir la dimension de la figure jusqu'à ce que celle-ci respecte les consignes puis l'affiche.

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

int main()
{int dim,nb,i,j,som;

    srand(time(NULL));
    do{
        printf("Entrez la dimension :");
        scanf("%d",&dim);
    }
    while(dim<3 || dim>21 || dim%2!=1 );

    for(i=1; i<=dim+2; i++) printf("*");
    printf("Som\n");

    for(i=1; i<=(dim+1)/2; i++){
        printf("*");
        for(j=1; j<=(dim+1)/2-i; j++)printf(" ");
        nb=rand()%10;som=0;
        for(j=1; j<=2*i-1; j++){
                printf("%d",nb);
                som +=nb;
        }
        for(j=1; j<=(dim+1)/2-i; j++)printf(" ");
        printf("* %d\n",som);
    }
    for(i=dim/2; i>=1; i--){
        printf("*");
        for(j=1; j<=(dim+1)/2-i; j++)printf(" ");
        nb=rand()%10;;som=0;
        for(j=1; j<=2*i-1; j++){
                printf("%d",nb);
                som +=nb;
        }
        for(j=1; j<=(dim+1)/2-i; j++) printf(" ");
        printf("* %d\n",som);
    }

    for(i=1; i<=dim+2; i++) printf("*");
    printf("\n");
}
```

Partie 2 (7 points) Idées de correction

2.1 Pointeurs (2 points)

Qu'affiche ce programme ?

```
#include <stdio.h>
int main(void)
{ int i,j;
   int *p1,*p2;
 i=7;
   j=i!=0;
   p1=&i;
   p2=&j;
   j++;
   i/=j;
   printf (" i=%d j=%d p1= %d p2 = %d", i,j,*p1,*p2);

   *p2 = i*j;
   *p1 *= j+1;
   p1=p2;
   printf("i=%d j=%d p1= %d p2 = %d", i,j,*p1,*p2+1);
```

i=3 j=2 p1= 3 p2 = 2

i=21 j=6 p1= 6 p2 = 7

2.1 Analyse d'un texte (3,5 points)

Ecrire un programme qui lit un texte (suite de caractère) terminé par le caractère # et détermine si ce texte est extrait d'un manuel littéraire, de comptabilité ou autre. On dira que l'extrait est : **littéraire** si au moins 70% de caractères sont des caractères alphabétiques (minuscule ou majuscule), **comptable** si au moins 60% des caractères sont des chiffres et **autre** sinon.

```
void main() {
    int nblettres =0, nbchiffres=0;
    char c;
    int nbc=0; // nb total de caracteres
    float pl,pc;    // pourcentages


    // lecture et comptage du nombre de chiffres et de lettres
    printf("rentrez votre texte d'au moins 1 caractère terminé par #  \n ");
    scanf("%c", &c);
    while (c!= '#') {
        if ((c>='a' && c<='z' )|| (c>='A' && c<='Z') ) nblettres++;
        else if (c>='0' && c<='9') nbchiffres++;
        nbc++;
        scanf("%c", &c);
        }

    // calcul des pourcentages et affichage (nbc différent de 0)
     pl = nblettres*100/(float)nbc;
    pc= nbchiffres*100/(float)nbc;
    if (pl >= 70 )
      printf ("\n %f pourcent de lettres: texte litteraire ", pl);
     else
      if (pc >= 60 )
         printf ("\n %f pourcent de chiffres : texte comptable ", pc);
      else
         printf ("\n type de texte inconnu");

}
```

2.2 Encadrement d'un réel par 2 entiers (1,5 points)

Ecrire un programme qui demande à l'utilisateur une valeur réelle x et affiche les 2 entiers les plus proches qui encadrent cette valeur. Par exemple,

pour x = 3.14519, le programme affichera 3 <= 3.14519 <= 4

pour x = -219.815 le programme affichera -220 <= -219.815 <= -219

Remarque importante : Le programme devra utiliser uniquement des conversions explicites (aucune conversion implicite ni fonctions prédéfinies).

```
void main() {
    int inf, sup;
    float  x;  // donnée

    printf("rentrez votre nombre SVP");
    scanf("%f", &x);
    if (x>=0) {
        inf = (int)x;
        sup = (int)(x+1);}
    else {
        inf = (int)(x-1);
        sup = (int)x;}

    printf ("\n %d <= %f  <= %d ", inf,x,sup);

      }
```

Partie 3 (7 points) Idées de correction

3.1 Table de multiplication (1,5 points)

Ecrire un programme qui saisit un nombre entier : si celui-ci est compris entre 1 et 9 alors le programme affichera la table de multiplication correspondante puis réitère le processus, sinon il affichera "Ce n'est pas dans les possibilités du programme, recommencez !". L'utilisateur tapera 0 (zéro) pour arrêter.

```
Quelle table ? (0 pour sortir) : 12
Ce n'est pas dans les possibilités
du programme, recommencez !
Quelle table ? (0 pour sortir) : 8
   8 * 0 = 0
   8 * 1 = 8
   8 * 2 = 16
   8 * 3 = 24
   8 * 4 = 32
   8 * 5 = 40
   8 * 6 = 48
   8 * 7 = 56
   8 * 8 = 64
   8 * 9 = 72
   8 * 10= 80
Quelle table ? (0 pour sortir) : 0
```

```
void main ()

{   int mult; //donné par l'utilisateur
    int i ;   // boucle d'affichage de la table

    do {
        printf("\n Quelle table de multiplication voulez-vous afficher, tapez 0
(zéro) pour sortir");
        scanf("%d", &mult);
        if (mult>9 || mult<0)
            printf("Ce n'est pas dans les possibilités du programme, recommencez
!");
        else
            if (mult > 0)
                for(i = 0; i < 10 ; i++)
                    printf("\n %d * %d = %d", mult, i, mult*i);
        }
    while (mult != 0);

}
```

3.2 Robot dans un cercle (5,5 points)

Un robot situé sur un quadrillage muni d'un repère orthonormé (O, i, j) se déplace d'une case à chaque pas dans une des quatre directions (haut, bas, droite, gauche) en fonction d'une valeur comprise entre 0 et 1. Cette valeur appelée « direction » est générée aléatoirement.
Au départ le robot est en O, à chaque pas, il se déplace vers la droite si la direction est inférieure strictement à 0.2, vers le haut si la direction est comprise entre 0.2 et 0.4, vers la gauche si la direction est supérieure entre 0.4 et inférieure à 0.7 et vers le bas si la direction est supérieure à 0.7.

Par exemple : si le robot est sur la case (3,4) et que la direction est 0.65 alors il se déplace d'une case vers la gauche et arrive en (2,4)

Le robot s'arrête dès qu'il sort du cercle de centre O et d'un rayon égal à r (valeur donnée par l'utilisateur).

1) Expliquez en quelques mots la manière dont vous allez calculer la valeur direction à l'aide de la fonction rand() qui retourne un entier positif ou nul.
2) Ecrire un programme qui permet de déplacer le robot à partir de sa position initiale, affiche ses coordonnées à chaque étape et affiche le nombre de déplacements lorsqu'il est arrivé sur le contour.
3) Ecrire un programme qui répète le processus précédent nb fois (nb donné par l'utilisateur) et qui détermine le nombre moyen de déplacements avant l'arrêt du robot ainsi que le pourcentage de fois où le robot sort du cercle par le haut (demi-cercle positif). Vous n'êtes pas obligés de tout réécrire.

Rappels : la fonction rand() retourne un entier positif ou nul.
L'équation d'un cercle centré en 0 et de rayon r est $x^2+y^2 = r^2$

```
int main () {
    float r ; // rayon du cercle
    int  x, y; // position initiale du robot
    int nb, i, nb_pos ;
    int  nb_cases;
    float moydep=0, pourcentage;   //
    float direction;


    printf("\n Introduire le nombre d'iteration\n");
    scanf("%d", &nb);
    printf("\n quelle est la taille du cercle ?\n");
    scanf("%f", &r);

    srand (time (NULL));

    //initialisations
    x=0; y=0;   // position du robot au centre du cercle
    nb_cases = 0;  // nombre de cases parcourues pour undeplacement
    nb_pos = 0 ;  // nombre de

    for (i=0; i<nb; i++) {
        while ((x*x + y*y)<=(r*r) {
            direction = (float)(rand()%101 )/100. ;

            if (direction < 0.3) {
                x++;
                nb_cases++;
            }
            if (0.3 <= direction && direction <= 0.5 ) {
                y++;
                nb_cases++;
            }
            if (0.5 < direction && direction <=0.7) {
                x--;
                nb_cases++;
            }
            if (0.7 < direction) {
```

```
            y--;
            nb_cases++;
        }

    }
    printf("\n les coordonnées de robot : (%d , %d)\n", x,y);
    printf("le nombre de cases pour ce deplacement : %d \n", nb_cases);

    // mise à jour des compteurs
    moydep += nb_cases;
    if (y>=0) nb_pos++;

    // remise à zero
    nb_cases=0;
    x=0; y=0;
}

moydep = moydep/(float)nb;   // cast optionnel
pourcentage = ((float) nb_pos*100)/(float)nb; // cast obligatoire

printf("\n  %.2f  cases en moyenne sont parcourues \n", moydep);
printf("le robot sort du cercle par le haut : %.2f %%   \n", pourcentage);


}
```