Université de Carthage
Ecole Supérieure de la Statistique et de l'Analyse de l'Information

Année Universitaire 2020/2021

Examen Période 1
Module : Algorithmique et programmation C I

Enseignantes : Mme Aïcha El Golli Jabbes/Sihem Mansour

Niveau d'études : 1ère année **Durée de l'épreuve :** 1h30

Remarques : *Les Documents, calculatrices, téléphone portable sont interdits.*
Veuillez rendre une copie propre et claire. Si la syntaxe d'une instruction est fausse alors la note est 0. La qualité de l'écriture et de la présentation sera prise en compte dans la note finale.

Exercice 1 (5 points)

Soit le programme suivant :

```
Procédure aff(d/r x : entier )

Début
    x ←x + 1;
    afficher(x+3);
 Fin
Fonction f(x : entier) : entier
Début
    x ← x + 1;

    afficher(x);
    si(x<10) alors
       retourne (x*3);
    sinon
      retourne(x/2) ;
    FinSi

Fin
Algorithme Essai
      y,t : entier ;
  Début
    y ← 9;
    aff(y);
    afficher("y= ", y );
    t ← f(y);
    afficher("y= ", y );
    afficher("t= ", t ) ;
 Fin
```

RAM

y=9- 10
t=5
x=9-10 (éliminée après exe de aff)
x= 10 11 (éliminée après exe de f

Ecran

13
y=10
11
y=10
t=5

```
int f(int x){
x+=1 ;
printf(" %i\n ",x) ;
if(x<10)
return (x*3) ;
else
return(x/2) ;
}
```

1. Exécuter le et donner la séquence des affichages produits (2,5pt)
2. Traduire la fonction f en langage C (2,5 pt)

Exercice 2 (6 points)

1. Soient les déclarations suivantes :

```
int n = 5, p = 9 ;
int q ;
float x ;
```

Quelle est la valeur affectée aux différentes variables concernées par chacune des instructions suivantes ?

a) `q = n<p ;  1(0,5pt)`
b) `q = n == p ;  0(0,5pt)`
c) `q = p % n + p ; 13 (1pt)`
d) `x = p / n ; 1.00 (0,5pt)`
e) `q = n *(p>n?n:p) ; 25 (0,5pt)`

2. Quelles sont les valeurs des deux variables i, j après exécution de ces instructions :

```
int i, j = 2;
i= ++j;
i++;
```

```
j--;
i= j--+j; ➔ i=3 et j= 1 (2pt)
```

3. L'expression : `X *= Y + 1 ;` est équivaut à :

a) `X = X * (Y + 1)` ➔ **la bonne réponse est a) (1pt)**

b) `X = X * Y + 1`

Exercice 3 (9 points) Centrer réduire

En théorie des probabilités et en statistique, une **variable centrée réduite** est la transformée d'une variable aléatoire par une application affine, de telle sorte que sa moyenne soit nulle et son écart type égal à un.

- Centrer une variable consiste à soustraire son espérance à chacune de ses valeurs initiales, soit retrancher à chaque donnée la moyenne (c'est ce qui s'appelle un centrage). Cela consiste simplement en un changement d'origine, qui place la moyenne de la distribution au point 0 de l'axe des abscisses.
- Réduire une variable consiste à diviser toutes ses valeurs par son écart type.

On désire écrire un algorithme qui construit un tableau de réels et décrivant une variable et qui permet ensuite de le centrer et réduire. L'algorithme, commenté par la suite, est le suivant :

Algorithme centreereduit
variables
moy, ecartT, t[100] **: réel ;**
n: **entier ;**
Début
init (t, n) ; *{procédure init décrite ci-dessous}*
moy←calcul_moy(t,n) ; *{fonction calcul_moy :* $\frac{\sum_{i=0}^{n-1} t[i]}{n}$ *}*
ecartT←calcul_ecart(t, n, moy) ;*{fonction calcul_ecart qui calcule l'écart type :* $\sqrt{\sum_{i=0}^{n-1} \frac{(t[i]-moy)^2}{n}}$ *}*
CentrerReduire(t, n, moy, ecartT) ; *{procédure CentrerReduire:* $t[i] = \frac{t[i]-moy}{ecartT}$*}*
Fin

1. Définir une procédure **init()** en algorithmique qui demande à l'utilisateur de saisir la valeur d'un entier ***n*** positif ou nul (cette valeur ne doit pas aussi dépasser la taille maximum du tableau, vérifier que la valeur saisie est bonne et redemander si nécessaire), et qui ensuite saisit les ***n*** éléments d'un tableau de réels.

(1.5pt)

```
Procédure init(d/r  t[] :réels, d/r  n : entier)
i : entier ;
Debut
Repeter
    Afficher(« donner un nombre positif inférieur à 100 ») ;
    Entrer(n) ;
Jusqu'à (n>=0)et (n<=100) ;
Pour(i de 0 à n-1) faire
    Entrer(t[i]) ;
fin
```

2. Définir la fonction **calcul_moy ()** en algorithmique qui calcule la moyenne arithmétique des éléments d'un tableau de réels : $\frac{\sum_{i=0}^{n-1} t[i]}{n}$
et qui renvoie la moyenne calculée. Cette fonction prend en paramètre le tableau et sa taille.

(2pt)

```
fonction calcul_moy( t[] :réel, n :entier) :réel
s :réel ;
i :entier ;
```

```
début
   s←0,0;
  Pour ( i de 0 à n-1) faire
        s←s+t[i] ;
FinPour ;
   Si(n<>0) retourne (s/n);
  Sinon retourne 0.0 ;
  Fin SI
Fin
```

3. Définir une fonction **calcul_ecart()**en algorithmique qui permet de calculer et retourner l'écart type des éléments d'un tableau de réels *:* $\sqrt{\sum_{i=0}^{n-1} \frac{|(t[i]\text{-}moy)|^2}{n}}$. Cette fonction prend en paramètre le tableau, sa taille et sa moyenne. Pour la racine carrée en algorithmique utilisez l'opérateur mathématique $\sqrt{\ }$.

(2pt)

```
fonction calcul_ecart(t[] :réel, n :entier, moy :réel) :réel
 d :réel ;
 i : entier;
début
   d←0,0 ;
     pour(i de 0 à n-1) faire
            d←d+(t[i]-moy)*(t[i]-moy);
     finpour
      Si(n<>0)  retourne (sqrt(d/n));
      Sinon retourne 0.0 ;
      Fin SI
fin
```

4. Traduire la fonction calcul_ecart() en langage C. Rappelons que le langage C a une bibliothèque prédéfinie <math.h> qui contient une fonction sqrt() et qui calcul la racine carrée d'un réel.

(2pt)

```
float calcul_ecart(float t[], int n, float moy)
{
   float d=0; int i;
     for(i=0;i<n;i++)
     {
        d+=(t[i]-moy)*(t[i]-moy);
     }
    if(n!=0) return (sqrt(d/n));
    else return 0.0;
}
```

5. Définir une procédure **centrerReduire()** en algorithmique qui consiste à soustraire la moyenne et diviser par l'écart type chaque élément du tableau : $t[i] = \frac{t[i]-moy}{ecartT}$. Cette procédure prend en paramètre le tableau, sa taille, sa moyenne et l'écart type.

(1,5pt)

```
procedure CentrerReduire(d/r  t[] :réel,  n : entier,  moy : reel,  ecartT : réel)
i :entier ;
début
    si (ecartT<>0)
            pour(i de 0 à n-1) faire
                    t[i]←(t[i]-moy)/ecartT;
             finpour ;
  finsi ;
Fin
```

Bon travail