

Examen partiel - semestre 1
Initiation à la programmation 2014/2015
CORRECTIONS
Électronique et Physique appliquée

*Tous documents
non électroniques autorisés
durée 1h*

Nom, prénom : **CORRECTIONS**

Spécialité :

- **Répondre sur le document.**
- Écrire vos programmes en langage C en respectant la syntaxe et l'indentation usuelle ;
- Soignez votre présentation (2 points bonus sont affectés à la présentation) ;
- Les questions sont relativement indépendantes ;
- Chaque question est notée sur 3 points.

1. Analyse de code

Qu'affiche le programme ci-dessous ?

```
#include<stdio.h>
int main (void) {
    int a = 4/3 ;
    float b = 4/3 ;
    float c = 4/3.0 ;
    printf ("a = %d, b = %f, c = %f\n", a, b, c) ;
    int d = 32%20 ;
    int e = 4, f = 18;
    int g = (e = f );
    printf ("d = %d, e = %d, f = %d, f = %x, g = %d\n", d, e, f, f, g) ;
    char h ='A' ; // pour info, le code ASCII de 'A' est 65
    h++ ;
    int i = (3*4==12) ? 8 : 9 ;
    printf ("g = %c, g= %d, h=%d (3*4==12) = %d \n", h, h, i, (3*4==12)) ;
}
```

a = 1, b = 1.000000, c = 1.333333
d = 12, e = 18, f = 18, f = 12, g = 18
g = B, g= 66, h=8 (3*4==12) = 1

2. Analyse de code

Qu'affiche le programme ci-dessous et pourquoi ?

```
#include<stdio.h>
int main (void) {
    double nb =0, nbTest =0, puiss = 1 ;
    int i=7 ;

    while (1) {
        nbTest=nb ;
        nb = nb + i/puiss ;
        if (nb==nbTest ) break ;
        puiss = puiss * 10 ;
    }
    printf ("nb = %f\n", nb) ;
}
```

nb = 7.777777

3. Programmation : boucles

Écrire un programme qui calcul les N premiers termes de la suite de Syracuse définie par la relation suivante :

*Si u_n est pair $u_{n+1}=u_n/2$
Sinon $u_{n+1} = 3*u_n + 1$*

U_0 et N seront demandés à l'utilisateur.

Voici une trace d'exécution :

```
Combien de termes à calculer : 15
Entrez U0 : 7
22 11 34 17 52 26 13 40 20 10 5 16 8 4 2
```

#include<stdio.h>

```
int main (void) {
    int n, un, un1, i ;
    printf ("Combien de termes à calculer : ") ;
    scanf ("%d", &n) ;
    printf ("Entrez U0 : ") ;
    scanf ("%d", &un) ;
    for (i=1 ; i<=n ; i++ ) {
        if (un%2==0)
            un1 = un/2 ;
        else un1=3*un+1 ;
        printf ("%d ", un1) ;
        un=un1 ;
    }
}
```

4. Programmation : tests

Écrire un programme qui tire un nombre au hasard compris entre 0 et 100, puis demande à l'utilisateur de le deviner. Lorsque l'utilisateur entre son nombre, la machine indique s'il est plus petit ou plus grand et invite l'utilisateur à essayer de nouveau jusqu'à ce qu'il trouve.

On utilisera la fonction

```
int rand ( void )
```

qui retourne un nombre aléatoire compris entre 0 et $2^{31}-1$

Voici une trace d'exécution :

```
Entrez le nombre à deviner : 50
Essayez avec un nombre plus grand
Entrez le nombre à deviner : 90
Essayez avec un nombre plus petit
Entrez le nombre à deviner : 86
Bravo !
```

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

int main (void) {
    int guess, nb ;
    srand( time(NULL)) ;
    guess = rand()%100 ;
    do {
        printf ("Entrez le nombre à deviner : ");
        scanf("%d", &nb) ;
        if ( nb < guess ) {
            printf ("Essayez avec un nombre plus grand\n");
        }
        else {
            if ( nb > guess ) {
                printf ("Essayez avec un nombre plus petit\n");
            }
        }
    }while (nb != guess) ;
    printf(" Bravo !\n");
}
```

5. Programmation : séries

Écrire le code de la fonction *ln1plus(x)* qui calcule $\ln(1+x)$ en utilisant le développement limité suivant :

$$\ln(1+x) = x - x^2/2 + x^3/3 - x^4/4 + x^5/5 + \dots$$

Le prototype de la la fonction est le suivant :

```
double ln1plus(double x) ;
```

La fonction donnera une valeur approchée à 10^{-10} près.

Voici le programme principal appelant la fonction :

```
int main (void) {  
    printf(" Le développement limité de ln(1 + 0.1) = %e \n", ln1plus(0.1) );  
}
```

Voici une trace d'exécution :

```
Le développement limité de ln(1 + 0.1) = 9.531018e-02
```

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

```
double ln1plus(double x)  {  
    double yn=0 ;  
    double yn1 = 0;  
    double n=1 ;  
    double expo = x;  
    do  {  
        yn = yn1 ;  
        yn1 = yn1 + expo/n ;  
        n=n+1 ;  
        expo=expo*(-x) ;  
    } while (fabs(yn1-yn) > 1e-10) ;  
    return (yn) ;  
}
```

6. Programmation : fonctions

Écrire la fonction `pgcd(...)` qui calcule le Plus Grand Commun Diviseur de 2 nombres. LE PGCD sera calculé par l'algorithme d'Euclide suivant.

$$PGCD(a, 0) = a$$

$$\text{Si } b \neq 0, PGCD(a, b) = PGCD(b, a \% b)$$

Voici le *main* permettant d'utiliser la fonction :

```
#include<stdio.h>
int main (void) {
    int a ,b ;
    printf ("Entrez 2 nombres dont vous voulez déterminer le PGCD : ") ;
    scanf ("%d %d", &a, &b) ;
    printf ("PGCD (%d, %d) = %d\n", a, b, pgcd(a,b)) ;
}
```

Et une trace d'exécution :

```
Entrez 2 nombres dont vous voulez déterminer le PGCD : 36 60
PGCD (36, 60) = 12
```

```
int pgcd (int a, int b) {
    int r ;
    if (b ==0) return (a) ;
    do {
        r=a%b ;
        a=b ;
        b=r ;
    } while(r!=0) ;
    return (a) ;
}
```