

ECUE «Introduction à la programmation »

Contrôle continu n°1 (CC1) 12 novembre 2018
sans document - durée 1 heure 30

Dans tous les exercices, les entrées clavier sont indiquées en caractères gras.

Exercice 1 (4 points)

Trouver et corriger les 8 erreurs du programme suivant :

```
#include <stdio.h>
int main() {
    int a;
    char b; c;
    printf("bonjour,\n");
    printf("tape un caractere: ");
    scanf("%c", &b);
    scanf("%c", c);
    printf("tu as tape %c\n", &b);
    printf("tape une valeur entiere: ");
    scanf("%d", &a);
    printf("tu as tape \n", a);
    printf("au revoir."\n);
    return 0;
}
```

Exercice 2 (5 points)

Ecrire un programme `exo2.c` permettant à l'utilisateur d'entrer 3 nombres entiers a , b , c et les affichant en ordre croissant, séparés par un `<` ou un `=` selon qu'ils sont différents ou égaux deux à deux. La sortie du programme doit correspondre aux cinq exécutions ci-dessous.

a ? 2	a ? 2	a ? 6	a ? 3	a ? 5
b ? 4	b ? 2	b ? 4	b ? 0	b ? 5
c ? 6	c ? 4	c ? 2	c ? 3	c ? 5
2<4<6	2=2<4	2<4<6	0<3=3	5=5=5

Exercice 3 (7 points)

Ecrire un programme `bridge.c` permettant à l'utilisateur de déterminer la carte gagnante d'un pli au jeu de bridge. Les entrées du programme sont une **couleur d'atout** (entier entre 0 et 3) et **quatre cartes** (quatre entiers entre 0 et 51). La sortie du programme est la **carte gagnante** (entier entre 0 et 51). Une **carte** est représentée par un nombre entier C entre 0 et 51. Sa **couleur** est le quotient de la division entière de C par 13 (entier entre 0 et 3). Sa **hauteur** est le reste de la division entière de C par 13 (entier entre 0 et 12).

Pour déterminer la carte gagnante, le programme détermine la couleur de la première carte : cette couleur s'appelle la **couleur demandée**. Si une carte de la couleur de l'atout existe, la carte

gagnante est la carte de hauteur la plus élevée de la couleur d'atout. Sinon, c'est la carte de hauteur la plus élevée de la couleur demandée. Les cartes de couleur autre que la couleur d'atout ou que la couleur demandée ne sont pas gagnantes.

Déroulement du programme : le programme demande la couleur d'atout. Si la couleur d'atout entrée par l'utilisateur n'est pas dans $\{0, 1, 2, 3\}$, le programme affiche une erreur et retourne -1 . Sinon il demande la première carte. Si la carte entrée n'est pas dans $\{0, \dots, 51\}$, le programme affiche une erreur et retourne -1 . Sinon il demande la deuxième carte. Etc jusqu'à la quatrième carte. Enfin, le programme affiche la carte gagnante et retourne 0 . Le programme respectera les entrées sorties des exécutions suivantes.

BRIDGE...	BRIDGE...	BRIDGE...
Atout ? 4	Atout ? 1	Atout ? 2
erreur: atout incorrect.	Carte 1 ? 33	Carte 1 ? 33
	Carte 2 ? 38	Carte 2 ? 38
BRIDGE...	Carte 3 ? 27	Carte 3 ? 16
Atout ? 1	Carte 4 ? 51	Carte 4 ? 51
Carte 1 ? 60	Couleur demandee : 2	Couleur demandee : 2
erreur: carte incorrecte.	CARTE GAGNANTE : 38	CARTE GAGNANTE : 38
BRIDGE...	BRIDGE...	BRIDGE...
Atout ? 1	Atout ? 1	Atout ? 0
Carte 1 ? 33	Carte 1 ? 33	Carte 1 ? 33
Carte 2 ? 38	Carte 2 ? 38	Carte 2 ? 38
Carte 3 ? 27	Carte 3 ? 27	Carte 3 ? 16
Carte 4 ? 35	Carte 4 ? 16	Carte 4 ? 5
Couleur demandee : 2	Couleur demandee : 2	Couleur demandee : 3
CARTE GAGNANTE : 38	CARTE GAGNANTE : 16	CARTE GAGNANTE : 5

Exercice 4 (4 points)

Donner la sortie du programme `exo4.c` ci-dessous, modifié en enlevant le `//` situé à gauche de la ligne de déclaration des variables `a, b, c`, correspondant à la couleur de votre copie.

```
// exo4.c
#include <stdio.h>
int main() {
    // int a=7, b=5, c=3; // copie rouge
    // int a=4, b=3, c=5; // copie bleue
    // int a=5, b=7, c=2; // copie verte
    // int a=6, b=3, c=4; // copie jaune
    printf("a+b*c=%d\n", a+b*c);
    printf("(a+b)*c=%d\n", (a+b)*c);
    printf("c/a = %d reste %d\n", c/a, c%a);
    a+=b+c;    printf("a=%d, b=%d, c=%d\n", a, b, c);
    b-=a+c;    printf("a=%d, b=%d, c=%d\n", a, b, c);
    c*=a*b;    printf("a=%d, b=%d, c=%d\n", a, b, c);
    a=a+(b++); printf("a=%d, b=%d\n", a, b);
    a=a+(++b); printf("a=%d, b=%d\n", a, b);
    return 0;
}
```