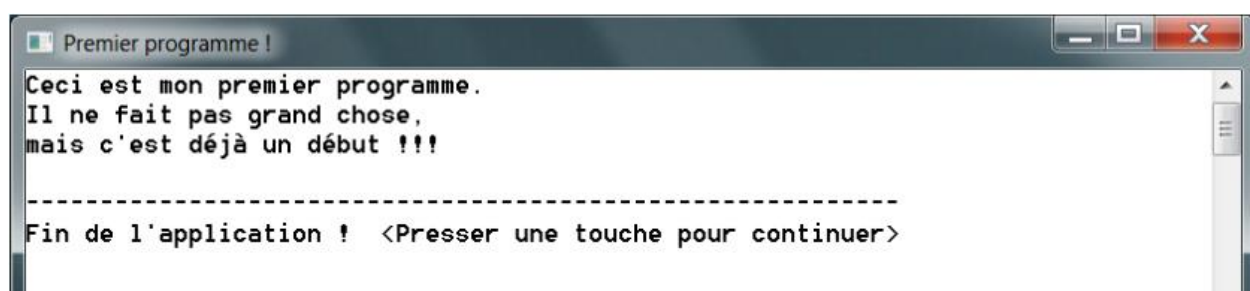


BASES

Exercice 1

Ecrire un programme qui affiche le résultat suivant à l'écran :

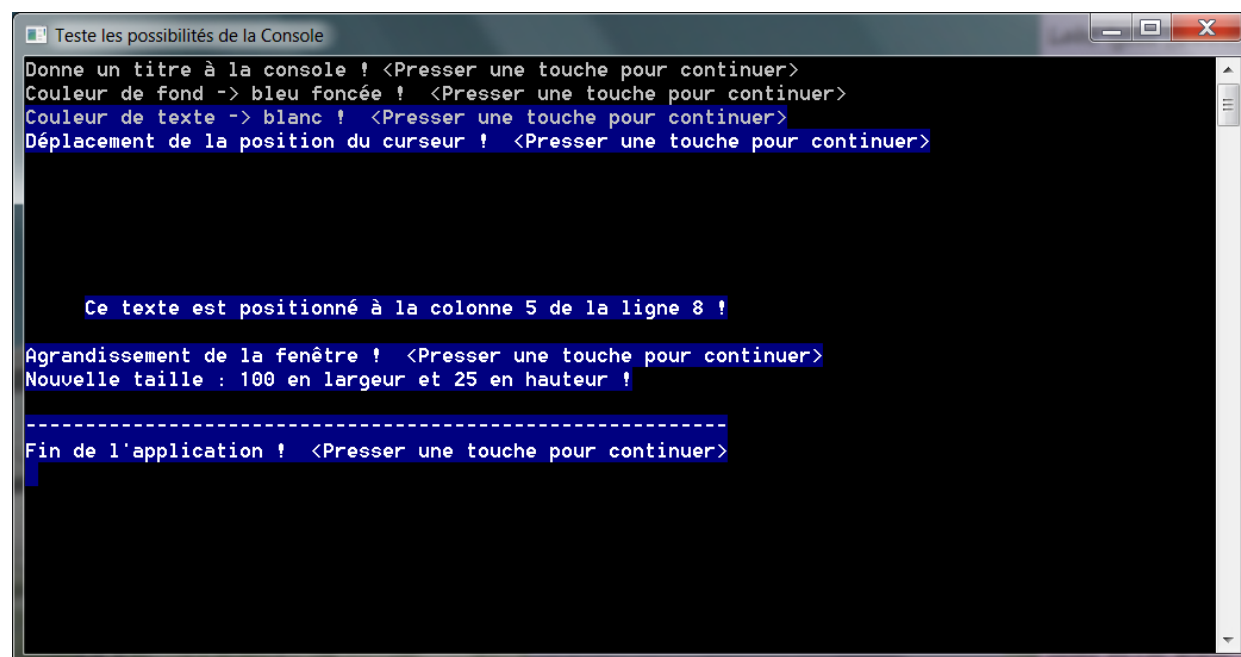


```
Premier programme !
Ceci est mon premier programme.
Il ne fait pas grand chose,
mais c'est déjà un début !!!

-----
Fin de l'application ! <Presser une touche pour continuer>
```

Exercice 1A – Test Console

Ecrire un programme afin de tester différentes fonctionnalités de la console comme la visualisation ci-dessous :



```
Teste les possibilités de la Console
Donne un titre à la console ! <Presser une touche pour continuer>
Couleur de fond -> bleu foncée ! <Presser une touche pour continuer>
Couleur de texte -> blanc ! <Presser une touche pour continuer>
Déplacement de la position du curseur ! <Presser une touche pour continuer>

      Ce texte est positionné à la colonne 5 de la ligne 8 !

Agrandissement de la fenêtre ! <Presser une touche pour continuer>
Nouvelle taille : 100 en largeur et 25 en hauteur !

-----
Fin de l'application ! <Presser une touche pour continuer>
```

Référence à lire : Chap. 1 point 1.11 Opérations d'entrée/sortie / p53

Exercice 2

LA CLASSE RANDOM

Constructeurs (Permettent d'instancier un ou des objets)

Nom	Description
Random	Initialise une nouvelle instance (nouvel objet) de la classe Random, à l'aide d'une valeur initiale par défaut qui est fonction du temps.
Random(Int32)	Initialise une nouvelle instance (nouvel objet) de la classe Random à l'aide de la valeur initiale spécifiée.

Méthodes

Nom	Description
Next	Retourne un nombre aléatoire non négatif.
Next(Int32)	Retourne un nombre aléatoire non négatif, inférieur au nombre maximal spécifié.
Next(Int32, Int32)	Retourne un nombre aléatoire figurant dans la plage spécifiée.
NextDouble	Retourne un nombre aléatoire compris entre 0,0 et 1,0.

Ecrire un programme qui génère aléatoirement quatre nombres aléatoire et affiche leur somme selon l'affichage ci-dessous.



```

Génère un nombre aléatoire entier : 0 <= nombre < 2147483647
Nombre 1 généré = 1422172838
Génère un nombre aléatoire entier : 0 <= nombre < 10
Nombre 2 généré = 4
Génère un nombre aléatoire entier : 5 <= nombre < 15
Nombre 3 généré = 6
Génère un nombre aléatoire réel : 0.0 <= nombre < 1.0
Nombre 4 généré = 0.319348562191868
-----
Somme des 4 nombres = 1422172848.31935
-----
Fin de l'application ! <Presser une touche pour continuer>
  
```

Référence à lire : Chap. 3 point 3.1.2 La classe Random / p166

Exercice 3

Ecrire un programme permettant de tester les différents types de données en C# :

```

C# Types et variables
Ce programme utilise une variable de chaque type du langage C#.
Il affiche également le valeurs minimales et maximales de ces variables.
>>> TYPES ENTIERS
    byte   : 234 valeur min : 0 valeur max : 255
    sbyte  : -102 valeur min : -128 valeur max : 127
    short  : -23432 valeur min : -32768 valeur max : 32767
    ushort : 43567 valeur min : 0 valeur max : 65535
    int    : -12342344 valeur min : -2147483648 valeur max : 2147483647
    uint   : 3343435657 valeur min : 0 valeur max : 4294967295
    long   : -1111111111 valeur min : -9223372036854775808 valeur max : 9223372036854775807
    ulong  : 23234332 valeur min : 0 valeur max : 18446744073709551615
>>> TYPES DECIMAUX
    float  : 1.333 valeur min : -3.402823E+38 valeur max : 3.402823E+38
    double : 0.3333333333333333 valeur min : -1.79769313486232E+308 valeur max : 1.79769313486232E+308
    decimal : 1.12312312312354 valeur min : -79228162514264337593543950335 valeur max : 79228162514264337593543950335
>>> TYPE BOOLEAN
    bool : True valeurs possibles : False , True
>>> TYPE CARACTERE
    char : Z valeur min =   valeur max : ?

-----
Fin de l'application ! <Presser une touche pour continuer>
Appuyez sur une touche pour continuer...
  
```

À noter que les valeurs mémorisées dans les différents types de données sont des nombres choisis par vos soins, directement intégrés dans le code source.

Référence à lire : Chap. 1 point 1.4 Types de données en C# / p24

Exercice 4

Ecrire un programme de test des fonctions mathématiques qui s'exécute comme ci-dessous :

```

C# Math & Cie
Ce programme teste et utilise quelques fonctions mathématiques du langage C#.
Pour cela, il demande d'introduire deux nombres décimaux a et b :
Valeur a : 3.4
Valeur b : 5.1

Nombre népérien --> 2.71828182845905
Nombre pi      --> 3.14159265358979

Elévation de a à la puissance b --> 513.502399028399
Racine carrée de a                --> 1.84390889145858

!!! Les angles doivent être donnés en radians dans les fonctions
Sinus de 45 degrés  --> 0.707106781186547
Cosinus de 0 degrés  --> 1
Tangente de 90 degrés --> 1.63317787283838E+16

-----
Fin de l'application ! <Presser une touche pour continuer>
Appuyez sur une touche pour continuer...
  
```

Référence à lire : Chap. 3 point 3.1.1 La classe Math / p163

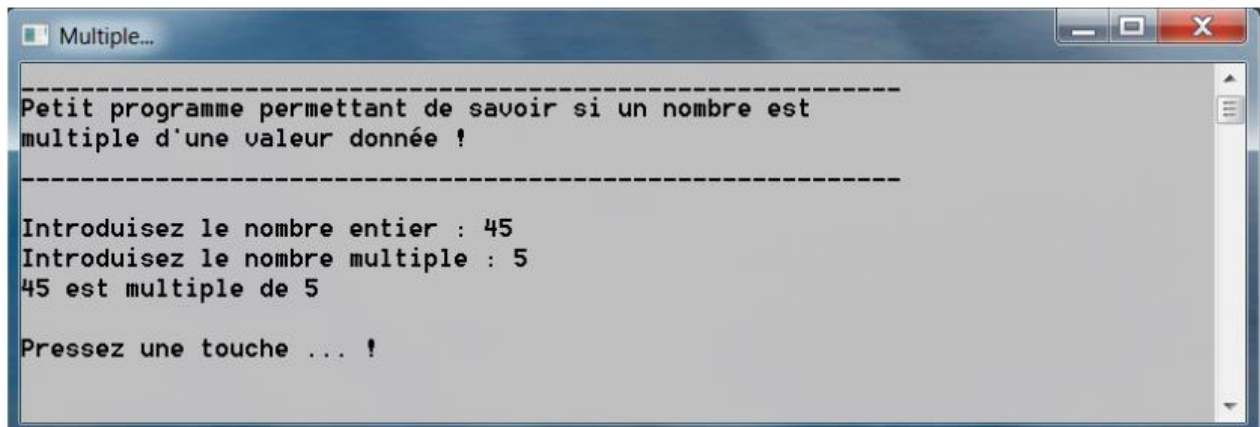
CONDITIONS

Exercice 5

Multiple

Ecrire un programme C# dont le but est de déterminer si un nombre (donné par l'utilisateur) est multiple d'une certaine valeur (donné par l'utilisateur).

Résultats :



```
Petit programme permettant de savoir si un nombre est
multiple d'une valeur donnée !

-----

Introduisez le nombre entier : 45
Introduisez le nombre multiple : 5
45 est multiple de 5

Pressez une touche ... !
```

Référence à lire : Chap. 1 point 1.12.1 Les opérateurs arithmétiques / p58

Exercice 6

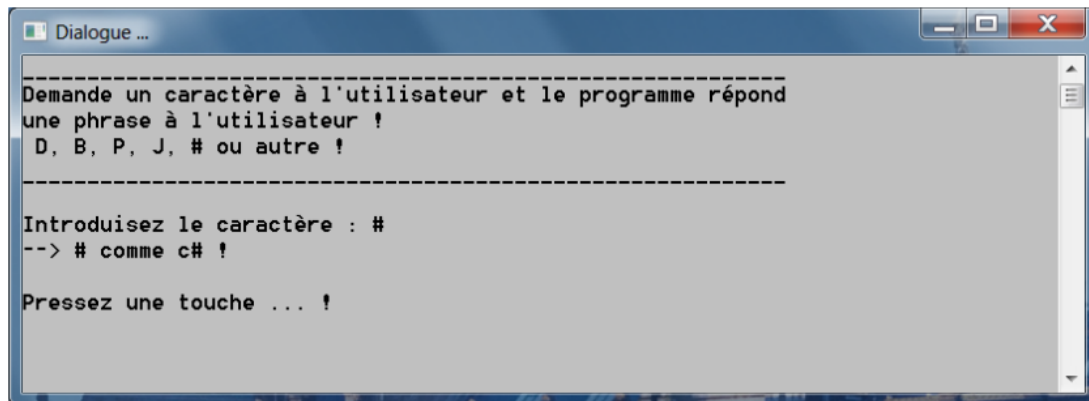
Dialogue

Ecrire un programme qui demande un caractère à l'utilisateur et affiche un message différent pour chaque caractère possible.

Selon le caractère tapé, le programme répondra les phrases suivantes

'd' ou 'D' alors affiche :	D comme début !
'b' ou 'B' alors affiche :	B comme b.a. ba !
'p' ou 'P' alors affiche :	P comme programmation !
'j' ou 'J' alors affiche :	J comme JAVA, ca m'éclate !
'#' alors affiche :	# comme C# !
autres lettres :	No comment !

Résultat :



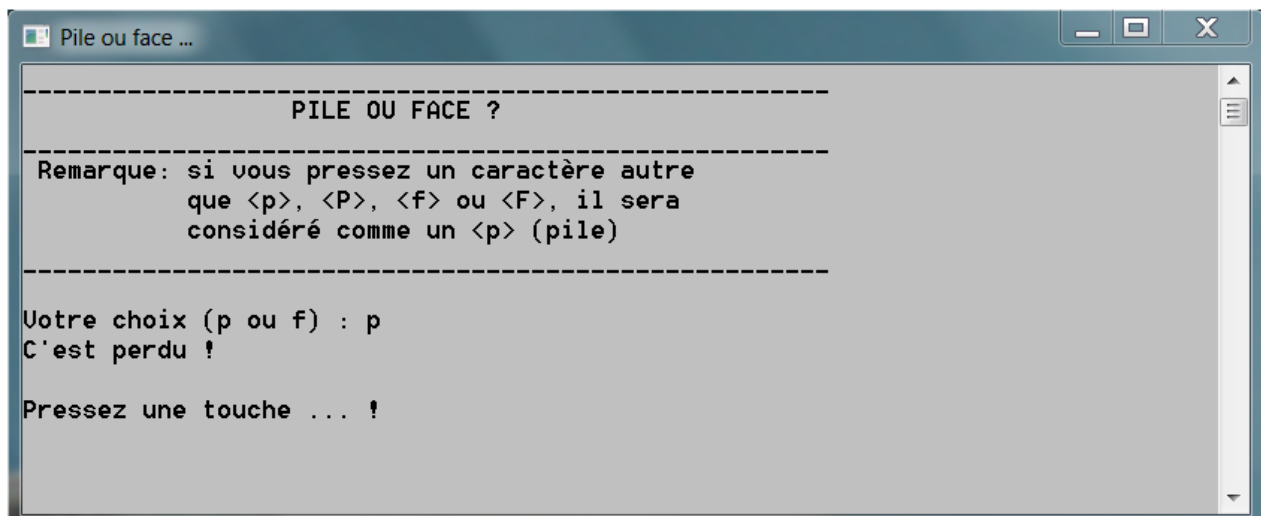
Exercice 7

Pile ou Face

Ecrire un programme qui demande à l'utilisateur : PILE ou FACE ?

Le programme choisi aussi aléatoirement PILE ou FACE. Si l'utilisateur a fait le même choix, alors le programme écrit : « C'est gagné ! » dans le cas contraire : « C'est Perdu ! ».

Résultat :



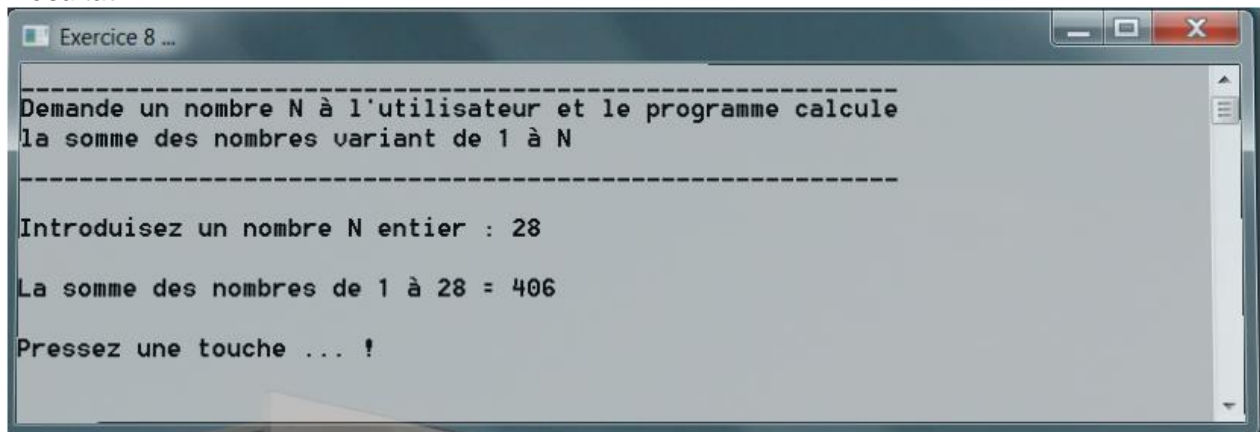
REPETITIONS - ITERATIONS

Exercice 8

Somme des entiers de 1 à n

Ecrire un programme qui calcule la somme des nombres entiers variant de 1 jusqu'à une valeur donnée par l'utilisateur. Le programme affichera à l'écran le résultat.

Résultat :



```
Exercice 8 ...
-----
Demande un nombre N à l'utilisateur et le programme calcule
la somme des nombres variant de 1 à N
-----
Introduisez un nombre N entier : 28
La somme des nombres de 1 à 28 = 406
Pressez une touche ... !
```

Exercice 9

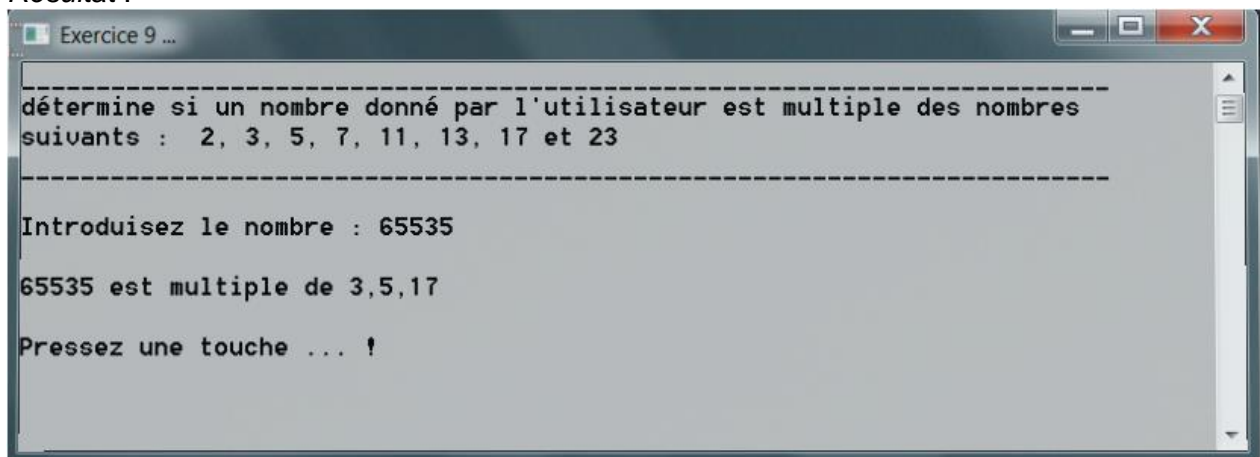
Multiple de 2, 3, 5, 7, 11, 13, 17 et 23

Ecrire un programme dont le but est de déterminer si un nombre (donné par l'utilisateur) est multiple des nombres suivants : 2, 3, 5, 7, 11, 13, 17 et 23.

Le résultat du programme sera donné selon la forme suivante :

« nombre donné » est un multiple de 2, 3, 5, 7, 11, 13, 17, 23

Résultat :



```
Exercice 9 ...
-----
détermine si un nombre donné par l'utilisateur est multiple des nombres
suivants : 2, 3, 5, 7, 11, 13, 17 et 23
-----
Introduisez le nombre : 65535
65535 est multiple de 3,5,17
Pressez une touche ... !
```

Exercice 10

Pile ou Face – amélioré ...

Transformer le programme de l'exercice 9 (Pile ou Face) en respectant les deux étapes suivantes :

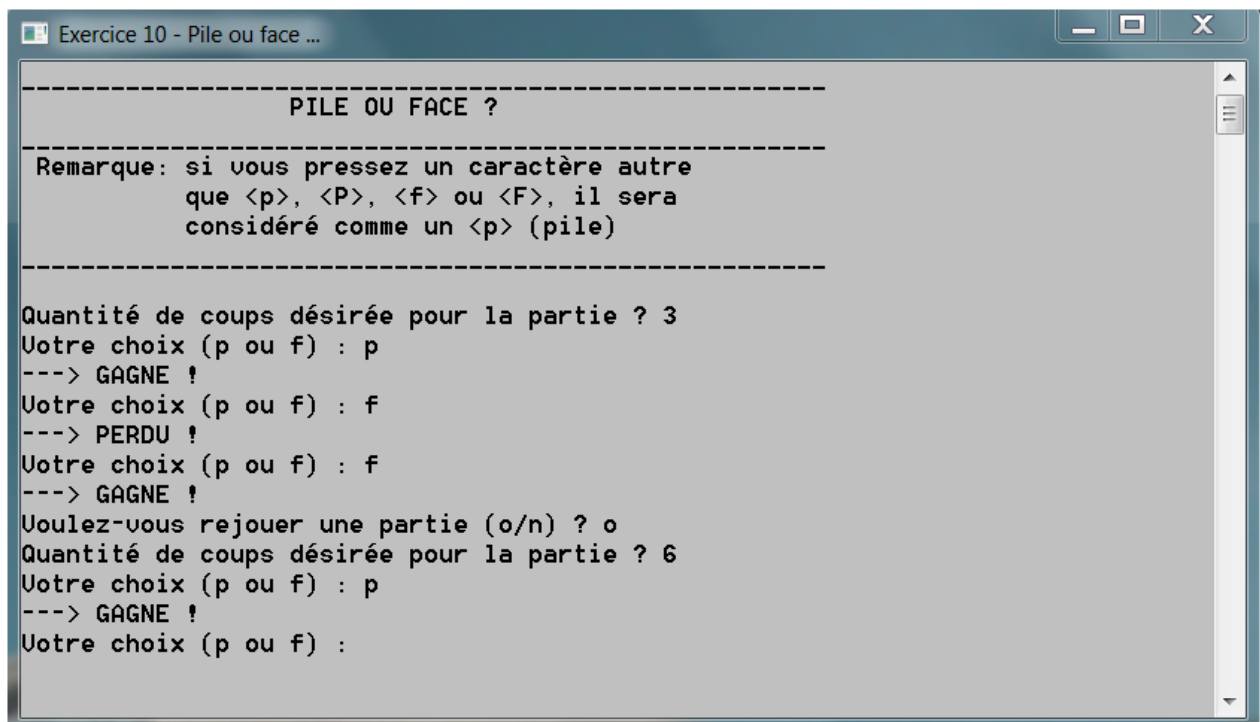
Étape 1

- Le programme doit permettre de jouer un nombre de coups défini dans une partie.
- Le programme affichera ensuite combien de fois le joueur a trouvé la réponse exacte.

Étape 2

- Le programme demandera à l'utilisateur s'il désire jouer à nouveau (faire une nouvelle partie).

Résultat :



```
Exercice 10 - Pile ou face ...
-----
PILE OU FACE ?
-----
Remarque: si vous pressez un caractère autre
           que <p>, <P>, <f> ou <F>, il sera
           considéré comme un <p> (pile)
-----

Quantité de coups désirée pour la partie ? 3
Votre choix (p ou f) : p
---> GAGNE !
Votre choix (p ou f) : f
---> PERDU !
Votre choix (p ou f) : f
---> GAGNE !
Voulez-vous rejouer une partie (o/n) ? o
Quantité de coups désirée pour la partie ? 6
Votre choix (p ou f) : p
---> GAGNE !
Votre choix (p ou f) :
```

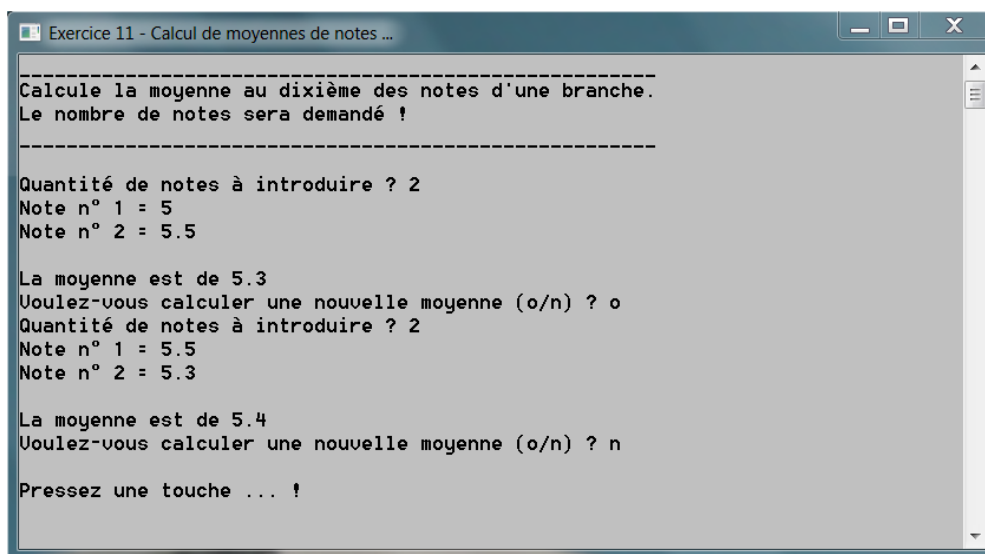
Exercice 11

Moyenne de notes

Ecrire un programme qui calcule la moyenne au dixième des notes d'une branche. Le nombre de notes à introduire sera demandée à l'utilisateur (ne pas utiliser de tableau... pas encore).

Le programme demandera également à l'utilisateur s'il désire calculer une autre moyenne.

Résultat :



```

-----
Calcule la moyenne au dixième des notes d'une branche.
Le nombre de notes sera demandé !
-----

Quantité de notes à introduire ? 2
Note n° 1 = 5
Note n° 2 = 5.5

La moyenne est de 5.3
Voulez-vous calculer une nouvelle moyenne (o/n) ? o
Quantité de notes à introduire ? 2
Note n° 1 = 5.5
Note n° 2 = 5.3

La moyenne est de 5.4
Voulez-vous calculer une nouvelle moyenne (o/n) ? n

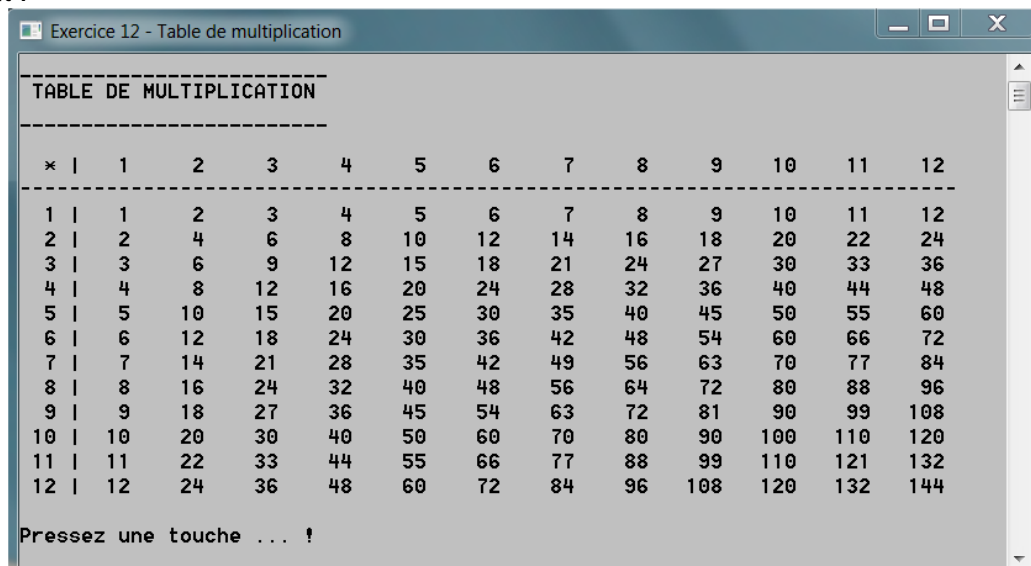
Pressez une touche ... !
  
```

Exercice 12

Table de multiplication

Ecrire un programme permettant d'afficher une table de multiplication selon le résultat suivant. Il n'y a aucun dialogue avec l'utilisateur.

Résultat :



×	1	2	3	4	5	6	7	8	9	10	11	12
1	1	2	3	4	5	6	7	8	9	10	11	12
2	2	4	6	8	10	12	14	16	18	20	22	24
3	3	6	9	12	15	18	21	24	27	30	33	36
4	4	8	12	16	20	24	28	32	36	40	44	48
5	5	10	15	20	25	30	35	40	45	50	55	60
6	6	12	18	24	30	36	42	48	54	60	66	72
7	7	14	21	28	35	42	49	56	63	70	77	84
8	8	16	24	32	40	48	56	64	72	80	88	96
9	9	18	27	36	45	54	63	72	81	90	99	108
10	10	20	30	40	50	60	70	80	90	100	110	120
11	11	22	33	44	55	66	77	88	99	110	121	132
12	12	24	36	48	60	72	84	96	108	120	132	144

Pressez une touche ... !

DIVERS

Calcul d'intérêts

Ecrire un programme qui va générer une table d'intérêts composés pour des taux de 5, 6, 7, 8, 9, 10 et 11 % et pour des durées de 1 à 15 ans.

Les valeurs de cette table représenteront le rapport C/c (**C**apital+intérê**t** / **c**apital investit).

Formule de l'intérêt composé :

$$C = c (1 + i)^n$$

où C = capital acquis
 c = capital investit
 i = taux d'intérêts
 n = nombres d'années

Nombres d'Armstrong

Un nombre d'Armstrong est caractérisé par le fait qu'il est égal à la somme des cubes des chiffres qui le composent.

Un nombre d'Armstrong est un nombre entier dont la somme du cube des chiffres le composant est égale au nombre.

Exemple: $153 = 1^3 + 5^3 + 3^3$

Ecrire un programme qui affiche l'ensemble des nombres d'Armstrong différents de 0 et 1 et inférieurs à 10000.

La pyramide

Sans utiliser un quelconque artifice d'affichage ; utilisation de boucles uniquement ; écrire l'algorithme affichant la pyramide suivante au centre de l'écran:

```
1
121
12321
1234321
123454321
12345654321
1234567654321
123456787654321
12345678987654321
```