*I2010 : langage C (2)*

# 1. Connaissances de base

En mathématiques, la fonction factorielle est définie comme suit :

0! = 1

n! = n\*(n-1)\*(n-2)\* ... \* 1 (pour n>0)

Ecrivez un programme qui affiche la factorielle d’un nombre. Il est demandé d’utiliser une boucle for pour effectuer le calcul et de ne pas définir, ni utiliser de fonctions[[1]](#footnote-1).

Différentes variantes vous sont demandées :

1. la valeur du nombre n est une constante dans le programme et il est demandé d’utiliser des nombres entiers (*int*) pour faire les calculs. Testez le programme avec les valeurs suivantes pour la constante :

0 ⇒

1 ⇒

5 ⇒

12 ⇒

13 ⇒

*Quels* *résultats sont non plausibles* ?

1. ajoutez dans votre programme l’affichage de la valeur de INT\_MAX et expliquez à partir de quelle valeur le calcul de la factorielle ne donne plus un résultat correct[[2]](#footnote-2).
2. modifiez le programme en utilisant une variable de type *double* pour les calculs intermédiaires et le résultat final. Testez votre programme avec les mêmes valeurs qu’au point 1 et discutez les résultats[[3]](#footnote-3).
3. modifiez votre programme du point 1 afin que la boucle for s’interrompe en affichant un message d’erreur lorsque le calcul de factorielle dépasse les capacités du type int.
4. modifiez le programme en utilisant une boucle while (les trois parties constituant la boucle for - initialisation, condition, adaptation - doivent y être présentes).
5. modifiez le programme afin que le test de dépassement de capacité soit intégré à la condition de la boucle while.
6. modifiez votre programme afin de calculer la factorielle de plusieurs nombres successivement. Les nombres sont formés d’un seul chiffre et sont introduits au clavier. Ils sont lus grâce à la macro getchar (attention, getchar lit des caractères, donc le code ASCII du chiffre[[4]](#footnote-4)). Le programme s’arrête quand il rencontre la marque de fin de fichier <Ctrl-D>. La vérification de dépassement de capacité lié au type int ne doit pas être vérifiée dans le cas d’un seul chiffre.

# 2. Lecture d’un nombre d’un seul chiffre

Reprenez le programme précédent qui calcule la factorielle de nombres d’un chiffre, lus grâce à la macro getchar.

Ajoutez-y tous les tests nécessaires afin que seule l’introduction de nombres d’un seul chiffre soit acceptée. Pensez à :

* vérifier que la ligne lue n’est pas vide
* vérifier que le caractère lu est compris entre 0 et 9
* vérifier que la ligne lue ne comprend que 2 caractères : le chiffre décimal directement suivi du return
* vider le buffer de lecture au cas où l’utilisateur aurait introduit plus de deux caractères

Testez votre programme avec le fichier entiers1.dta.

Pour ce faire, si votre exécutable est factoriel, tapez la commande suivante pour rediriger l’entrée standard de votre programme vers le fichier entiers1.dta :

./factoriel < entiers1.dta

Vérifiez que votre programme traite correctement les données (comparez vos résultats avec le fichier entiers1.dta.out).

# 3. Lecture, avec la macro getchar, de nombres de 1 ou 2 chiffres

Ecrivez une autre version de ce programme qui calcule la factorielle de nombres de un ou deux chiffres lus grâce à la macro getchar.

Ajoutez tous les tests nécessaires afin que seule l’introduction de nombres de maximum deux chiffres soit acceptée.

Assurez-vous que le résultat renvoyé par votre programme soit toujours correct.

Testez votre programme avec le fichier entiers2.dta et vérifiez les résultats avec le fichier entiers2.dta.out.

# Bonus : Puissances entières

Ecrivez un programme qui calcule la puissance d’un entier par un autre entier (sans utiliser la fonction pow de la librairie *math.h*). Les nombres entiers sont lus au clavier grâce à la macro getchar. Ils sont composés de 1 ou 2 chiffres et peuvent être négatifs.

Rappelons que :

a(-b) = 1/ab.

Assurez-vous que votre programme traite correctement les cas limites :

* 00=1
* a0=1
* 0(-b)=∞ si b>0

1. L'affichage du contenu d'une variable entière se fait grâce à la fonction printf avec le format %d comme dans l'exemple printf("Le contenu de la variable i est %d\n", i) (cf. eSyllabus chap. 6) [↑](#footnote-ref-1)
2. La même problématique informatique est à l’origine du « [bug de l'an 2038](https://fr.wikipedia.org/wiki/Bug_de_l%27an_2038) » qui concerne les logiciels utilisant la représentation POSIX du temps. [↑](#footnote-ref-2)
3. L'affichage du contenu de la partie entière d'une variable réelle se fait grâce à la fonction printf avec le format %.0f comme dans l'exemple printf("Le contenu de la partie entière de la variable double x est %.0f\n", x) (cf. eSyllabus chap. 6) [↑](#footnote-ref-3)
4. Consultez le chapitre 6 de l’eSyllabus ou tapez la commande man getchar dans un terminal pour accéder à la page de manuel de la fonction. [↑](#footnote-ref-4)