MÈTODES NUMÈRICS I

Grau de Matemàtiques. Curs 2016/17. Semestre de tardor

PRÀCTICA 1

Es proposa de fer uns quants programes en llenguatge C que serveixin per a comprovar les **limitacions de les variables de tipus numèric**:

- Les variables *int* prenen valors en un **rang limitat**.
- Les variables *float* i *double* també tenen **limitació de rang** i, a més, **precisió limitada** o **finita**.

Nota: Aquesta primera pràctica també ha de servir per a **repassar l'edició, la compilació i l'execució de programes en C**:

- Es treballa des del sistema operatiu *linux*.
- Un editor que hi ha instal·lat és *vi* (de fet *vim*).
- Instrucció bàsica de compilació: *gcc -c -ansi -Wall arxiu.c*
- Muntatge: *gcc arxiu1.o arxiu2.o ... -o nom_programa.exe -lm*
- Execució: *./nom_programa.exe*

El nivell de C necessari per a aquest curs és el dels **apunts**:

Aubanell, Bañeres, Font, Romano: Elements de programació en llenguatge C. Apunts. Departament de Matemàtica Aplicada i Anàlisi de la UB.

Exercici 1 [Rang de valors de les variables *int*]

Feu un programa que només usi variables de tipus *int* i faci els càlculs següents:

(i) 10^i , $i = 1, 2, 3, \ldots, 12$.

(ii) 2^i , $i = 1, 2, 3, \ldots, 32$.

(iii) Vagi incrementant en 1 una variable inicialitzada a 0, fins que no augmenti més!

(iv) Vagi disminuint en 1 una variable inicialitzada a 0, fins que no disminueixi més!

(v) $i!$, $i = 1, 2, 3, \ldots, 35$.

En tots els càlculs anteriors s'obtenen inconcruències matemàtiques. Sabeu explicar-les?

Compareu el que heu obtingut amb les constants *INT_MIN* i *INT_MAX* que hi ha a la llibreria *limits.h*

Exercici 2 [Rang de valors de les variables *float* i *double*]

Programeu (i), (ii) i (v) de l'exercici anterior però usant variables de tipus *float* (i després de tipus *double*). Feu que l'índex i prengui valors més grans, fins que es produeixin **overflows**.

Fent que l'índex i prengui valors negatius, comproveu que també es produeixen **underflows**.

Nota. Feu escriure tots els valors amb molts decimals. Així veure-ho també que, abans de produir-se un *overflow* o un *underflow*, ja es perd precisió (no totes les xifres que s'escriuen són correctes).

Compareu els vostres resultats amb els valors de les constants *FLT_MIN*, *FLT_MAX*, *DBL_MIN* i *DBL_MAX* que hi ha a la llibreria *float.h*, les quals contenen els valors positius màxim i mínim, *normals* que es poden guardar.

Exercici 3 [Error de representació]

Llegiu un valor real i escriviu-lo amb molts dígits. Comproveu que, sovint, no escriu el mateix que ha llegit. Exemple clàssic: 0.1.

Es diu que es treballa amb *precisió finita*, o que *existeix un error de representació*. Feu-ho tant en una variable *float* com en una variable *double*.

Per tal de trobar una fita de l'error de representació en el cas *float*, feu el següent: doneu un mateix valor a una variable *double* x i a una variable *float* a. Considereu el valor a com una aproximació del valor exacte x i calculeu l'error relatiu $e = |(a - x)/x|$. Feu això sistemàticament (per exemple per a $x = 0.01, 0.02, 0.03, \ldots, 1000.00$, o per a altres conjunts de valors) i trobeu una fita pràctica de l'error relatiu (o sigui, el màxim de tots).

Exercici 4 [Epsilon de la màquina]

Si es va calculant $1+u$ per a valors positius de u cada vegada més petits, arriba un moment que el resultat és 1.

Definició. S'anomena **epsilon de la màquina** al mínim valor real positiu $u > 0$ que verifica que $1 + u$ dóna diferent de 1.

Feu un programa que calculi aquest valor, tant en *float* com en *double*, usant $u = 2^{-i}$, $i > 0$. Compareu els vostres resultats amb les constants *FLT_EPSILON* i *DBL_EPSILON* que hi ha a la llibreria *float.h*.

Té alguna relació la fita de l'exercici anterior amb l'epsilon de la màquina?