

Práctica 6

Francisco Rivera Álvarez

1. Introducción

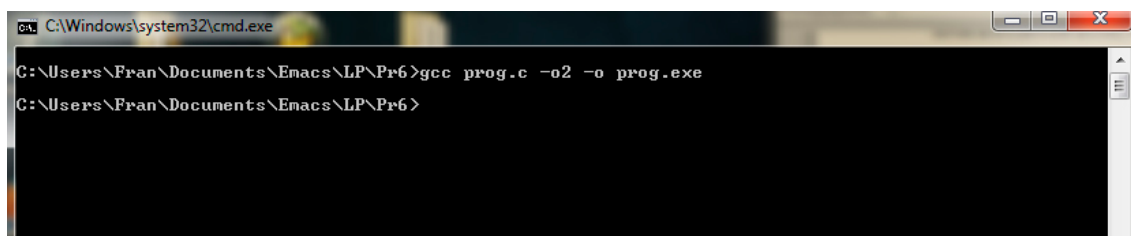
El siguiente programa fue diseñado para hallar mediante simulaciones de Montecarlo el número PI. Originalmente estaba escrito en fortran 77, pero se ha traducido a C. Explicaré como se debe compilar y ejecutar para poder usar en aquellos ordenadores que tenga un compilador de C.

2. Compilado y linkado

Escribiremos en la Shell o Consola el comando:

- `gcc prog.c -o2 -o prog.exe`

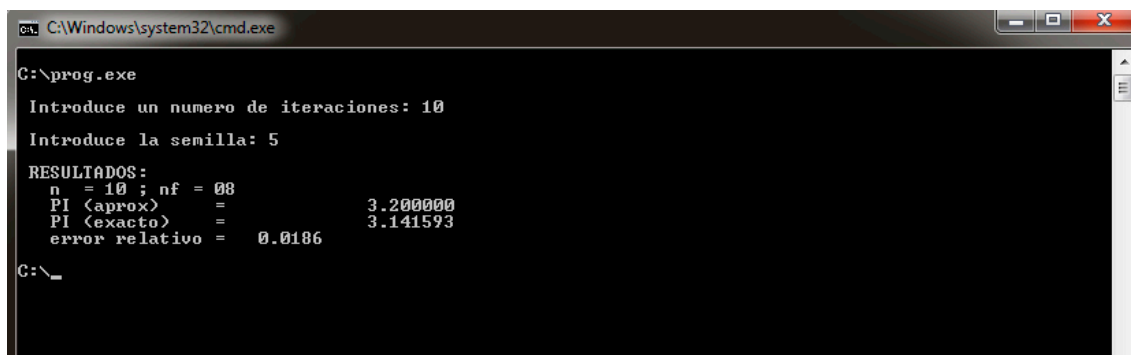
Para que el programa GCC encuentre el archivo prog.c debemos colocarnos en la carpeta donde se encuentra el archivo, mediante los comandos `cd` y `dir` (Windows) o `ls` y `cd` (Linux).



```
C:\Windows\system32\cmd.exe
C:\Users\Fran\Documents\Emacs\LP\Pr6>gcc prog.c -o2 -o prog.exe
C:\Users\Fran\Documents\Emacs\LP\Pr6>
```

3. Ejecución

Una vez compilado podemos ejecutar el programa. Una forma sería usando la consola escribiendo el nombre del programa (`prog.exe`) y pulsamos Enter (en Linux sería `./prog.exe`). También podemos buscar la carpeta y abrirlo con el explorador, luego haríamos click sobre el programa.



```
C:\Windows\system32\cmd.exe
C:\>prog.exe
Introduce un numero de iteraciones: 10
Introduce la semilla: 5
RESULTADOS:
n = 10 ; nf = 08
PI <aprox> = 3.200000
PI <exacto> = 3.141593
error relativo = 0.0186
C:\>
```

Una vez ejecutado, el programa nos pide dos datos. El primer dato que debemos introducir es el número de iteraciones. Cuantas más iteraciones mejor será la precisión. El segundo dato es la semilla, valor que usa el ordenador para generar número aleatorios. Introduzca un valor entero en ambos casos.

4. Comentarios

La aproximación depende tanto del nº de iteraciones como del número aleatorio.

```

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

#define PI 3.14159265358979324

//este programa calcula el valor de Pi mediante un metodo de Monte-Carlo
int main(int argc, char *argv[])
{
    int n;                //numero de iteraciones
    unsigned int seed;     //semilla para cambiar el nÂ° aleatorio generado

    while(0 < 10){
        printf("\n Introduce un numero de iteraciones: ");
        scanf("%d", &n);
        if(n > 0) break;
    }

    while(0 < 10){
        printf("\n Introduce la semilla: ");
        scanf("%d", &seed);
        if(seed > 0) break;
    }

    //inicializamos la generaciÃ³n de NÂ° aleatorios
    srand(seed);

    //simulaciÃ³n
    int nf=0;
    int i;
    for (i = 0; i < n; i++) {
        double x = (double) rand()/RAND_MAX;
        double y = (double) rand()/RAND_MAX;
        if(x*x+y*y < 1.)
            nf++;
    }

    double pi_aprox = 4. * (double) nf / (double)n;
    double error = (pi_aprox - PI)/PI;

    //imprimimos los resultados
    printf("\n RESULTADOS: \
        \n    n = %02d ; nf = %02d \
        \n    PI (aprox)      = %20.6f\
        \n    PI (exacto)      = %20.6f\
        \n    error relativo = %8.4f",
        n,nf,pi_aprox,PI,error);

    //para cerrar el programa el usuario debera pulsar una tecla
    fflush(stdin); getchar();

    return 0;
}

```