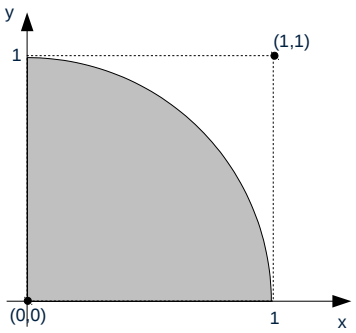
Práctica 2. Números pseudoaleatorios

Fecha de entrega: 22 de Enero de 2010

Introducción

El objetivo de esta práctica es el cálculo de una aproximación al número π mediante el *método de Montecarlo*. Para ello se considera la porción del primer cuadrante del plano delimitada por una circunferencia de radio 1 con centro en el origen de coordenadas. Esta región corresponde al área sombreada en la siguiente figura.



El procedimiento consiste en la selección de varios puntos (x, y) al azar, con $0 \leq x, y \leq 1$ (es decir, dentro de la zona delimitada por líneas discontinuas). Para cada punto, diremos que se ha producido un *acierto* si el punto ha caído dentro de la zona sombreada. Esto ocurrirá, a su vez, si la distancia entre el punto y el origen de coordenadas es menor o igual que 1.

Se cumple la siguiente relación entre las proporciones del número de aciertos y el área sombreada:

$$\frac{\textit{número de aciertos}}{\textit{número total de puntos}} \approx \frac{\textit{área sombreada}}{\textit{área total}}$$

En nuestro caso, dado que el área de la zona sombreada es $\pi/4$ y el área total es 1, se tiene:

$$\frac{\textit{número de aciertos}}{\textit{número total de puntos}} \approx \frac{\pi}{4} \Longrightarrow \pi \approx 4 \cdot \frac{\textit{número de aciertos}}{\textit{número total de puntos}} \tag{1}$$

Cuanto mayor sea el número de puntos, mayor será la precisión obtenida.

Procedimiento

El programa a realizar consistirá en la implementación de este proceso: se selecciona una cantidad fija de puntos y se contabiliza el número de aciertos. A partir de estos y utilizando la fórmula (1) se calcula el valor aproximado de π. Un ejemplo de ejecución de este programa se muestra a continuación:

```
Número de intentos: 1000000
Número de aciertos: 785416
PI = 3.14166
```

Para ello, procederemos del siguiente modo:

1. Crear una función con el siguiente prototipo:

```
void punto_aleatorio(double &x, double &y)
```

Esta función ha de seleccionar aleatoriamente un punto con coordenadas dentro del rango $0 \ldots 1$. Cada coordenada será devuelta en el correspondiente parámetro pasado por referencia.

Recuerda que la función `rand()` devuelve un número *entero* comprendido entre 0 y el valor de la constante `RAND_MAX`. Para obtener un número *decimal* comprendido entre 0 y 1 tan sólo es necesario dividir el valor devuelto por `rand()` entre `RAND_MAX`.

2. Codificar una función que calcule la distancia entre dos puntos (x_1, y_1) y (x_2, y_2):

   ```
   double distancia(double x1, double y1, double x2, double y2)
   ```

   Para ello se debe utilizar la siguiente fórmula:

   $$distancia = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

   La función `sqrt` (definida en `cmath`) calcula la raíz cuadrada de un número.

3. Utilizar las funciones descritas anteriormente para escribir un programa que genere varios puntos aleatoriamente y determine cuántos de ellos son aciertos. A partir de esto, mostrar por pantalla el valor aproximado de π utilizando (1).

Evaluación

La corrección de la práctica es un requisito indispensable. Además, se valorará positivamente la claridad del código (abundancia de comentarios, nombres de identificadores adecuados, etc...).