Computación 2013-1
Tarea 3

Entrega: Domingo 7 de octubre de 2012
por correo electrónico a Iker:* icasik@ciencias.unam.mx

Recuerda utilizar los dos puntos : y la sangría de cuatro espacios para los ciclos y las condiciones.

1. Un número primo es un número entero que sólo es divisible por sí mismo y uno. Escribe un programa que encuentre el 10001-ésimo número primo. Por un punto extra demostrar y encontrar cuál es el mayor entero que hace falta descartar para asegurarse que un número es primo.

2. El número más pequeño que es divisible por todos los números del 1 al 10 es 2520. Encuentra el número más pequeño que es divisible por todos los números del 1 al 20.

3. Ordenar un conjunto de datos es necesario en muchas circunstancias. Por ejemplo, ordenar los productos en una tienda por precio o por popularidad; ordenar una lista de nombres por calificaciones o por edad, etc. Un algoritmo fácil de implementar es el de ordenar una lista usando inserción. Dada una lista desordenada L_d con k números $n_0 \cdots n_{k-1}$, se construye una lista ordenada L_o, en orden ascendente, como sigue:

Para cada $i \in \{2, 3, \ldots, k-1, k\}$

1. Comenzando desde el primer elemento $L_d[0] = n_0$, se considera una lista L_i de longitud i.
2. Dado el último elemento de L_i, i.e. $L_i[i-1]$, se compara su valor con el del elemento anterior $L_i[i-2]$. Si $L_i[i-1] > L_i[i-2]$ entonces esos dos elementos están ordenados, en caso contrario se intercambian y se continúa de la misma manera hasta insertar al elemento $L_i[i-1]$ en la posición que le corresponde. Al finalizar este procedimiento, la lista L_i está ordenada.

Observaciones: Cada vez que se realiza el paso 2., se obtiene una lista ordenada al principio de L_d. Después de realizar el paso 2. para todos los valores de i, se ha transformado a L_d en L_o.

Implementa una función `ordenar()` que ordene la siguiente lista de números:

$$[-3, 0, 25, 1, 3.14, 7, -24, 1]$$

Nota: Recuerda que en Python una lista se define como `a=[1,2,3]` y un elemento de la lista es `a[indice]`. Para una lista de n elementos, los índices van desde 0 hasta $n-1$.

4. Buscar un valor v en un conjunto de datos puede ser una tarea difícil, sobre todo si se tiene una enorme cantidad de datos. Las cosas se simplifican, si los datos están ordenados de alguna forma. Un algoritmo sencillo y eficiente que permite buscar en una lista L de k números, ordenada (en orden ascendente), es el de búsqueda binaria. Este algoritmo consiste en comparar v con el valor del número m que está justo a la mitad de la lista. Si $v == m$ entonces encontramos a v en la lista. Si $v > m$ entonces buscamos en la mitad derecha de la lista. De lo contrario buscamos en la mitad izquierda. En la mitad correspondiente repetimos el mismo procedimiento hasta determinar si v se encuentra en la lista o no.

Implementa una función `bbinaria()` que busque el valor 0 en la lista ordenada del problema anterior.

*Recuerda enviar únicamente un archivo con extensión .py por cada problema. Cualquier observación o texto debe ir dentro de cada programa como comentario.