Criptografía

Práctica 1: Potencia modular , Logaritmo discreto y Factorización.

__TrollScience__

Sergio Balbuena Pantigas
Jesús Carabaño Bravo
Pablo Caro Martín
Francisco Miguel Dios Buitrago

Descripción del Software

Este programa ha sido implementado sobre el lenguaje de programación **Python** en su más reciente versión **3.2.2** y explota las funcionalidades que esta brinda, por lo que será necesario utilizar un interprete en su versión 3.x para lograr su correcto funcionamiento.

El contenido del programa se ha desglosado en tres secciones. La primera incluye las funcionalidades cuya implementación era obligatoria, como la **potencia modular**, el **logaritmo discreto**, la **tabla comparativa** entre los dos anteriores y el test de **primalidad probabilística**; la segunda sección reune las funcionalidades opcionales, que en esta práctica consisten en la factorización de grandes números en primos y que reune los algoritmos de factorización de **fuerza bruta**, de **Rho de Pollard**, de **Fermat** y de **Strassen**. La última sección que añadimos presenta algunas herramientas útiles para la práctica concreta y en este caso la única herramienta añadida consiste en un **generador de números impares** con tamaño a elegir que pretende facilitar las pruebas de los algoritmos.

Propiedades del Software

Las características más destacables de nuestro programa son la **facilidad de uso**, la **eficiencia** de ejecución y la **simplicidad del código** elaborado. Veamos por qué las destacamos:

- **Facilidad de Uso.** No existen más opciones de las requeridas ni menos de las necesarias y aunque la interfaz del tipo consola sea simple es más que suficiente para su labor. Utilizar un algoritmo solo requiere de dos sencillas interacciones con el teclado.
- **Eficiencia.** Es de sobra conocido que el lenguaje Python se encuentra a un nivel de abstracción superior a los clásicos lenguajes de programación de alto nivel. Este hecho repercute en la velocidad de ejecución de los programas generados, siendo normalmente más lentos que con otros lenguajes. Sin embargo, a manos de un buen programador esas deficiencias pueden solventarse e incluso es posible conseguir mejores rendimientos haciendo uso de las utilidades avanzadas que Python ofrece. Este ejemplo podemos apreciarlo en el logaritmo discreto, que consigue excelentes tiempos de ejecución gracias al uso de estructuras con eficiencias de búsqueda alta.
- **Simplicidad del código.** Esta propiedad no es visible al usuario medio pero es interesante conocer que, por ejemplo, con Python no es necesario controlar el overflow de sus números, pues internamiente gestiona cualquier tamaño que sea necesario. Esta y otras características dotan de simplicidad al código y hacen posible una reutilización más agil y sencilla.

Manual de Uso

La utilización del programa es muy simple. En primer lugar ha de ejecutarse sobre algún interprete Python 3.x, sin argumentos:

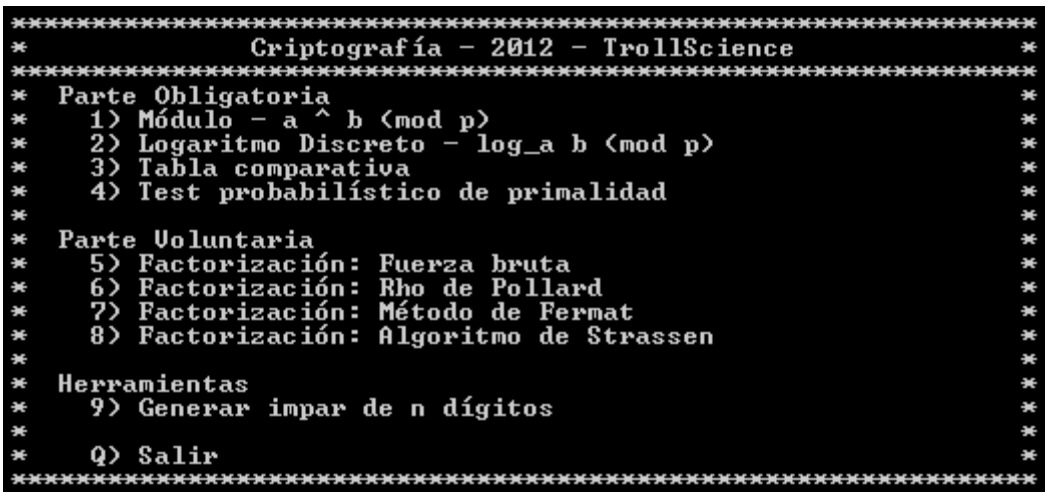
> python P1.py

Según la instalación de Python, puede ser necesario llamarlo así:

> python3 P1.py

Al lanzar la aplicación aparecerá un menú de elección en el que podrá decidirse la funcionalidad a utilizar mediante el teclado. Cuando se elija algún algoritmo el programa solicitará un número sobre el que aplicarlo. Una vez introducido el número el programa pedirá al usuario los valores requeridos, realizará los calculos necesarios y comunicará las salidas pertinentes, así como el tiempo dedicado.

Nota: Para más facilidad de uso, el test de primalidad permite dos tipos de entradas en el número de rondas. Un número entero (p.e. 10) indica el número de rondas del algoritmo. Un número con o sin decimales y acabado en '%', sin espacio (p.e. 99.9999%), indica el porcentaje de probabilidad que se desea, el programa ajustará las rondas necesarias.



Comparativa: Potencia-Logaritmo

Primo utilizado	Cifras	Tiempo de potencia modular (segundos)	Tiempo de logaritmo discreto (segundos)
46199	5	1.999855041503906e-05	0.00020000934600830078
629243	6	1.999855041503906e-05	0.0010001659393310547
7715177	7	3.000020980834961e-05	0.0019998550415039062
82376809	8	3.000020980834961e-05	0.00800013542175293
756065159	9	2.9997825622558592e-05	0.023000001907348633
4093082899	10	3.000020980834961e-05	0.06500005722045898
735773424149	12	2.0000934600830078e-05	0.47099995613098145
5746175430239	13	3.000020980834961e-05	1.49399995803833
25569934528771	14	3.000020980834961e-05	3.746000051498413
740745508071383	15	3.000020980834961e-05	25.07200002670288