INF-221 Algoritmos y Complejidad, 2018-2

Tarea 1

Profesor: Diego Arroyuelo
Ayudantes: Manuel Weitzman, Ignacio Tampe
`manuel.calquin.14@sansano.usm.cl`,
`ignacio.tampe@sansano.usm.cl`

Fecha de Entrega: 2 de noviembre de 2018
Plazo máximo de entrega: 2 días.

Reglas del Juego

La presente tarea debe hacerse en grupos de 3 personas. Toda excepción a esta regla debe ser conversada con los ayudantes **ANTES** de comenzar la tarea. No se permiten de ninguna manera grupos de más de 3 personas. Las tareas deben compilar en los computadores que se encuentran en el laboratorio B-032 (LDS). Deben usarse los lenguajes de programación C o C++. Se recomienda compilar en el terminal usando `gcc archivo.c -o output -Wall` (en el caso de lenguaje C) o `g++ archivo.cpp -o output -Wall` (en el caso de C++). Si usa C++, está permitido emplear estructuras de datos de la `stl`, en caso de ser necesarias.

Strings Secretos

Alicia y Bob quieren comunicarse por la red sin que Eva pueda leer sus mensajes. Para esto han inventado un esquema que les permite enviar mensajes ocultando su contenido. Dado un mensaje M de largo n y conformado por los caracteres $m_1 m_2 m_3 \dots m_n$, se genera un mensaje cifrado C simplemente revolviendo los caracteres de M, de modo que el mensaje se convierte en $c_1 c_2 c_3 \dots c_n$. Así ya no importa que Eva logre ver los mensajes, ya que para ella parecerá que Alicia y Bob sólo se envían *strings* aleatorios.

Para que el receptor pueda des-cifrar el mensaje, debe permutar sus caracteres. La permutaciones se generan seleccionando los caracteres de C de izquierda a derecha (c_1 hasta c_n) e insertándolos en todas las posiciones posibles (de izquierda a derecha) de un nuevo *string* P. Esto genera $n!$ posibles permutaciones, las cuales deben ser enumeradas. Para decidir el valor de P, se debe conocer el índice de la permutación que le corresponde, al cual denotaremos como K (de *key*, o contraseña). Por ejemplo, para el mensaje cifrado C = LAOH y contraseña K, se debe seleccionar la K-ésima permutación de C, como se muestra en la Tabla 1.

Caracter a insertar				
L	A	O	H	K
L	AL	OAL	HOAL	1
			OHAL	2
			OAHL	3
			OALH	4
		AOL	HAOL	5
			AHOL	6
			AOHL	7
			AOLH	8
		ALO	HALO	9
			AHLO	10
			ALHO	11
			ALOH	12
	LA	OLA	HOLA	13
			OHLA	14
			OLHA	15
			OLAH	16
		LOA	HLOA	17
			LHOA	18
			LOHA	19
			LOAH	20
		LAO	HLAO	21
			LHAO	22
			LAHO	23
			LAOH	24

Tabla 1: Permutaciones para “LAOH”.

Es decir, existe una función $F(C, K) = P \equiv M$ que es capaz de convertir un mensaje cifrado C en uno de texto plano P igual al original. Por ejemplo

$$F(\text{LAOH}, 13) \equiv \text{HOLA}$$

Alicia y Bob implementaron dicha función y coordinaron de antemano una serie de valores de K para usar. Por desgracia, una plaga de ratones provocó una falla eléctrica en los servidores que alojaban el código, y necesitan su ayuda para re-implementar la función F, de la cual sólo recuerdan que usaba la técnica de *backtracking* y era eficiente en tiempo así como uso de memoria, requerimientos imprescindibles.

Formato de Entrada

El ingreso del input se hará a través de la entrada estándar (`stdin`). La primera linea posee un entero positivo $T < 2^{32}$ indicando la cantidad de casos de prueba que le siguen. Por cada caso de prueba hay dos líneas. La primera contiene un *string* de largo n (tal que $0 < n \leq 32$). La siguiente contiene un índice de permutación k $\left(\text{tal que } 1 \leq k \leq \text{mín}\left\{n!,\ 2^{128} - 1\right\}\right)$. Un ejemplo de caso de prueba es el siguiente:

```
4
USM
4
ABCD
10
CDAB
10
QAZCDETGBMJUYHNVFRWSXIKLOP
165423
```

Para probar su programa de una mejor manera, ingrese los datos de entrada con el formato indicado en un archivo de texto (por ejemplo, el archivo `input.txt`). Luego, ejecute su programa desde la terminal, redirigiendo la entrada estándar como a continuación:

```
./tarea1 < input.txt
```

de esta manera evita tener que entrar los datos manualmente cada vez que prueba su programa, y evita errores.

Formato de Salida

La salida se hará a través de la salida estándar (`stdout`). Para cada caso se debe imprimir una línea con el *string* correspondiente a la k-ésima permutación. El output para el caso de ejemplo es

```
MUS
BDAC
DBCA
IXSWRFVNHYPKUOJMLBGTEDCZAQ
```

Bonus por Eficiencia

Se darán bonus de a las 3 tareas que ejecuten más rápido y resuelvan correctamente el problemas. Dicho bonus se podrá usar en cualquiera de las tareas del curso, y será aplicado de manera automática procurando maximizar el promedio. Los bonus corresponden a los indicados en la Tabla 2.

Lugar	Bonus
1	30
2	20
3	10

Tabla 2: Bonus para las 3 tareas más rápidas.

Entrega de la Tarea

La entrega de la tarea debe realizarse enviando un archivo comprimido llamado

`tarea1-apellido1-apellido2-apellido3.tar.gz`

(reemplazando sus apellidos según corresponda) en el sitio Aula (`https://aula.usm.cl`) del curso, a más tardar el día 2 de noviembre de 2018, a las 23:55:00 hrs (Chile Continental), el cual contenga:

- Los archivos con el código fuente necesarios para el funcionamiento de la tarea.
- `NOMBRES.txt`, Nombre y ROL de cada integrante del grupo.
- `README.txt`, Instrucciones de compilación en caso de ser necesarias.
- `Makefile`, Instrucciones para compilación automáticas.

Restricciones y Consideraciones

- Por cada día de atraso en la entrega de la tarea se descontarán 20 puntos en la nota.
- El plazo máximo de entrega es 2 días después de la fecha original de entrega.
- Pueden programar la tarea en C o C++ según ustedes consideren conveniente.
- Las tareas deben compilar en los computadores que se encuentran en el laboratorio B-032. **Las tareas que no compilen no serán revisadas y serán calificadas con nota 0, sin excepciones**.
- Por cada *Warning* en la compilación se descontarán 5 puntos.
- Si se detecta **COPIA** la nota automáticamente sera 0 (CERO), para todos los grupos involucrados. El incidente será reportado al Jefe de Carrera.
- No respetar el formato de input/output implica un descuento considerable (al menos la mitad del puntaje máximo) en la evaluación de la tarea. Esto incluye no usar la entrada y/o salida estándar.
- La prolijidad, orden y legibilidad del código fuente es obligatoria. Habrá descuentos de hasta 20 puntos si alguno de estos ítemes no se cumple.