**Explicación del AE & AEE en el programa**

­Para poder determinar las ecuaciones que podrán utilizarse para cifrar y descifrar con los datos ingresados, es menester saber qué es lo que hace —de manera general— cada uno de estos algoritmos en este apartado.  
El **AEE** permite al programa encontrar al inverso multiplicativo del valor de α (α-1), ya que es necesaria para establecer y calcular la función de descifrado al quitarle los corchetes. No obstante, el programa se asegura primero de que dicho valor existe a través de la obtención de *MCD (n, α)* usando el **AE**. Si el resultado de la expresión equivale a 1, el AE procederá, por lo que es virtualmente imposible que falle una vez que se ha establecido la única restricción de su uso. Veamos cómo operan:

El AE se desarrolla mediante operaciones con listas dentro de un bucle While.

Una de ellas contiene a n & α, la lista “modules”, la cual es encargada de almacenar los valores de los residuos de una serie de divisiones. La razón tras su agregación a la lista es que el AE expresará un número a partir de otro en sucesión, donde n & α son los primeros dos números, respectivamente.

Asimismo, la lista “naturalDiv” almacena cocientes de dichas divisiones, los cuales son utilizados para generar ecuaciones nuevas dentro del algoritmo de Euclides.

Como se sabe que el algoritmo finaliza hasta que el residuo termine siendo 0, ello es declarado como condición del ciclo y se realizarán tantas iteraciones sean necesarias hasta que se cumpla dicha condición.

En cada una de las iteraciones, se realiza una división natural entre el número actual de la lista “modules” y el siguiente (recordemos que inicia con dos números). El resultado de la división es almacenado en “naturalDiv”. Una vez que se ha detenido el arreglo, se estipula que el penúltimo dato de la lista “modules” es el máximo común divisor de los valores n y α. En caso de ser uno, significa que ambos números son coprimos y —por consecuencia— son candidatos a generar ecuaciones de cifrado y descifrado.

Adicionalmente, se notará que “naturalDiv” contiene valores cuyo uso no es necesario para el funcionamiento. Son vestigio de una función adicional planeada, la cual consistía en mostrar el desarrollo operacional de los cálculos. Está presente para AE, pero fue descartada para AEE dada la dificultad de la implementación de un sistema así de completo.

Hablando de dicho, a continuación, se muestra el funcionamiento y explicación del algoritmo AEE:

Para la implementación del algoritmo, se requiere modificar los valores de n y α, por lo que se hacen copias dado que los originales serán utilizados para generar las ecuaciones de cifrado y descifrado.

Asimismo, recordemos que el algoritmo nos lleva a una expresión que tiene la forma *1 = ax + by*, donde x & y son coeficientes cuyo valor aumenta conforme el algoritmo se desarrolla, por lo que este desarrollo se almacena progresivamente en listas homónimas. x & y no son coeficientes cualesquiera. Son conocidos bajo el nombre de Coeficientes de Bézout y ya están asociados a una fórmula que permite calcularlos conforme el AE se desarrolla normalmente.

* *xi = xi-2 – qi(xi-1)*
* *xi = xi-2 – qi(xi-1)*

Son las fórmulas en cuestión. Por convención, los valores que tienen los subíndices negativos son:

* xi-2 = 1; xi-1 = 0
* yi-2 = 0; yi-1 = 1

Por lo que dichos valores se inician en las listas previamente a la implementación del AEE.

Lo que se implementa es, básicamente, otro AE donde estas fórmulas se están aplicando en segundo plano. Una vez que el AE termina, las fórmulas se encargan de conseguir los valores finales para x & y. Son comprobados en la ecuación planteada inicialmente y luego el coeficiente de interés es procesado para su implementación en las fórmulas de descifrado.

El Algoritmo de Euclides es un algoritmo matemático que se utiliza para calcular el Máximo Común Divisor de dos números a & b a través de la descomposición de uno a partir del otro:  
a = bq + r, donde q equivale a a/b (a dividido por b) y r es el residuo de la operación previamente mencionada.

Para continuar el algoritmo, se aplican las ecuaciones a = b & b = r, y se reinicia el desarrollo. El algoritmo se da por concluido cuando r equivale a 0, y el valor previo que tuvo r antes de terminar es considerado el MCD de ambos números a & b.

Por otro lado, el Algoritmo Extendido de Euclides dice que, de existir MCD(a, b), existe también una ecuación que cumple que ax + by = MCD(a, b).

El Algoritmo Extendido trabaja mediante la inversión de los pasos del AE común, mediante el cual se aplican propiedades aritméticas para encontrar una combinación lineal que cumpla con ax + by = MCD(a, b). La ecuación previa recibe el nombre de Identidad de Bézout, y tanto a x como a y se le conocen como Coeficientes de Bézout.

Trabajando regresivamente, se han encontrado ecuaciones que calculan los coeficientes que satisfacen dicha ecuación, las cuales son:

* xi = xi-2 – qi(xi-1)
* xi = xi-2 – qi(xi-1)

Donde q es el cociente que resulta de hacer a/b repetidamente, al igual que el Algoritmo de Euclides convencional. De esta manera, es posible desarrollar el algoritmo normalmente mientras se van calculando los Coeficientes de Bézout. Al tener que acceder a datos previos a la implementación, se asume que:

* xi-2 = 1; xi-1 = 0
* yi-2 = 0; yi-1 = 1

Al aplicar las ecuaciones conjuntamente con el Algoritmo de Euclides, se calculan los coeficientes al final simultáneamente.