



Complejidad computacional 2023-II

Practica 2

Acosta Arzate Rubén 317205776
López Miranda Ángel Mauricio 317034808



Esquema de codificación:

Nuestro esquema de codificación para los ejemplares es el mismo que la practica pasada.
Nuestras cadenas son de la forma:

```
#k
$S
\\c1 %c2 %... %cn
```

Donde nuestra k es nuestro entero para saber con que K-cubierta estamos trabajando
S es nuestro conjunto universo
y nuestras $c_i \in C$ con $1 \leq i \leq n$ donde C es un conjunto de subconjuntos de S

Usamos los caracteres # para determinar quien es nuestra k, \$ para determinar quien es nuestro conjunto S *universo*, \\ para reconocer donde empieza nuestro conjunto C y por ultimo % para separar cada $c_i \in C$

Es importante que se siga ese orden en la codificación, i.e, el símbolo # en la primera linea del archivo, y sin espacios el valor de k.

ejemplo: #5

En la segunda linea debe estar el símbolo \$ seguido de una cadena sin espacios con los elementos del conjunto S separados por comas.

ejemplo: \$1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23

En la tercera linea debemos tener \\ seguido de una cadena sin espacios que contenga comas para separar los elementos de cada subconjunto c_i y con el símbolo % para marcar el inicio de otro subconjunto c_i .

ejemplo \\1,2%1,2,3,4,5,6,7,8,9,10%11,12,13,14,15%3,4,5,6%7,8,9%16,17,18,19,20%21,22,23

Como se puede inferir la implementación es sensible a espacios en blanco y saltos de linea por lo que si no se sigue lo antes mencionado el algoritmo regresará mensajes como: “El valor de K no se encontró en la codificación”.

Por otro lado la codificación de los archivos que contienen el certificado no requieren de caracteres especiales sin embargo si se quiere generar un archivo de certificado manualmente se debe tener en cuenta que es una cadena de 1's & 0's sin espacios y sin comas, la cadena evidentemente debe de tener una longitud de tamaño igual a la cardinalidad del conjunto C del ejemplar en concreto y un numero de 1's igual a la K del ejemplar, si la cadena no cumple con estas especificaciones será un certificado invalido, impidiendo la verificación de pertenencia del ejemplar al lenguaje.

Esquema de certificación:

Ya que nuestro problema a resolver es encontrar k subconjuntos de S tal que al unirnos nos de como resultado el conjunto S en su totalidad y por el sistema de codificación conocemos quien es nuestra k, nuestro conjunto S y nuestros subconjuntos c_i entonces nuestros certificados serán de la forma:

$e_1e_2\dots e_n$ siendo $e_i \in \{0, 1\}$ y $n = |C|$ y nuestro certificado contendrá k 1s y esto representara los subconjuntos que verificaremos si la unión de estos es igual a S , es decir
 $c_i \cup c_j \dots \cup c_k = S$

Algoritmo de generación aleatoria de certificados:

Nuestro algoritmo es relativamente sencillo pues simplemente generamos un arreglo de n 0s donde $n = |C|$ y generando aleatoriamente un numero entre 0 y $n-1$ colocamos k 1s en los índices obtenidos aleatoriamente y simplemente para asegurar que sean k 1s cuando se colocara un 1 en el índice del arreglo dado se revisa si en este índice ya hay un 1 si ya hay pues genera un numero aleatorio diferente.

Por ultimo simplemente concatenamos cada elemento del arreglo con una cadena y con esto obtenemos el certificado.

Si quisiéramos calcular la complejidad de este algoritmo pues tenemos que recibimos 2 parámetros principales siendo n y k donde k es nuestro entero para una K -cubierta y n son la cantidad de subconjuntos de C

Donde básicamente creamos un arreglo de n 0s y lo recorremos para colocar k 1s por lo que en el peor de los casos recorremos k veces nuestro arreglo de n elementos por lo tanto tenemos

$$O(n,k)=kn$$

Algoritmo de verificación:

Para nuestro algoritmo de verificación lo que hicimos es que recibiendo nuestro certificado y nuestro ejemplar codificado verifique si los conjuntos dados son igual al conjunto universo S .

Como funciona pues básicamente tenemos una cadena de longitud $n = |C|$ digamos cer y una cadena que representa a un ejemplar que al codificarla con el mismo método de la practica 1, obtenemos a k , S y a C

Ya con estos parámetros revisamos en que índice de nuestra cadena cer tenemos 1s y tomamos los subconjuntos de C en esos índices y los concatenamos, luego nuestra cadena la convertimos en una lista y eliminamos repetidos y comprobamos si esta lista obtenida es igual a S .

```
verificador(cer,S,C):
cadena=""
for i in range(0,len(cer)):
    if i==1:
        cadena+=C[i]

lista=split(cadena,",")
eliminaReps(lista)

if lista==S:
    return true
else
```

- return false

Donde len es un método que nos calcula la longitud de una cadena, split es un método que nos separa en una lista una cadena tomando como delimitador el carácter en el segundo parámetro y eliminaReps un método que al pasarle una lista nos elimina repetidos

Si quisiéramos ver la complejidad de este algoritmo seria:

Primero tenemos un ciclo donde concatenamos a la cadena los subconjuntos de C, ya que sabemos que la longitud de C es n, y situamos el peor caso como agregar a todo C esto nos queda O(n)

Luego nos queda eliminar el método split, este recibe 2 argumentos y lo que hace es crear una lista vacía, y recorre la cadena hasta encontrar ”,” si lo encuentra agrega todo lo anterior como un elemento de nuestra lista, por lo que como la acción mas larga es recorrer la cadena nos queda igual O(n)

Posteriormente eliminaReps recibe de argumento una lista y lo que hace es que crea una lista vacía l y va recorriendo cada elemento de la lista y verifica si ese elemento que tomo esta en L y si no lo agrega de tal manera que podemos decir que para revisar si hay algún repetido en el peor de los casos sea $l = |lista|$ recorre a la lista l veces por lo que eso es $O(l^2)$ y como el peor caso de la lista es que fuera todo C, pues nos queda como $O(n^2)$

Por ultimo checamos que lista == S para checar eso supondremos que lo hace por el peor caso que es comparando cada elemento de lista y checando que sea el mismo que S por lo que en el peor caso recorres todo S y toda la lista por lo que esto nos queda sea $m = |S|$ $O(2m)$ que se reduce a $O(m)$

Por lo que tendríamos $O(m,n) = O(n) + O(n) + O(n^2) + O(m)$ lo que se reduce a $O(m, n) = n^2 + m$

lo cual sigue siendo de orden polinomial por lo que nuestro algoritmo verificador cumple

Resumen de las pruebas ejecutadas

Para el ejemplar 1:

K=3

S={1,2,3,4,5,6,7,8,9,10,11,12,13,14,15}

C={ {1,2} {1,2,3,4,5,6,7,8,9,10} {11,12,13,14,15} {3,4,5,6} {7,8,9} }

Para el ejemplar 2:

K=4

S={1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20}

C={ {1,2} {1,2,3,4,5,6,7,8,9,10} {11,12,13,14,15} {3,4,5,6} {7,8,9} {16,17,18,19,20} }

Para el ejemplar 3:

K=5

S={1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23}

C={ {1,2} {1,2,3,4,5,6,7,8,9,10} {11,12,13,14,15} {3,4,5,6} {7,8,9} {16,17,18,19,20} {21,22,23} }

```
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$ python verificador.py
Escribe con que ejemplar quieres trabajar
Puedes escoger entre los siguientes archivos:
ej1
ej2
ej3
Archivo:ej1
Escribe el archivo donde tienes el certificado
Puedes escoger entre los siguientes archivos:
cer1
cer2
cer3
cer4
cer5
Archivo:cer1
El numero de elementos de S es: 15
El numero de subconjuntos de C es: 5
Valor de K = 3
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10']
[['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15']]
No, los conjuntos tomados no cumplen para ser una cubierta para S
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$
```

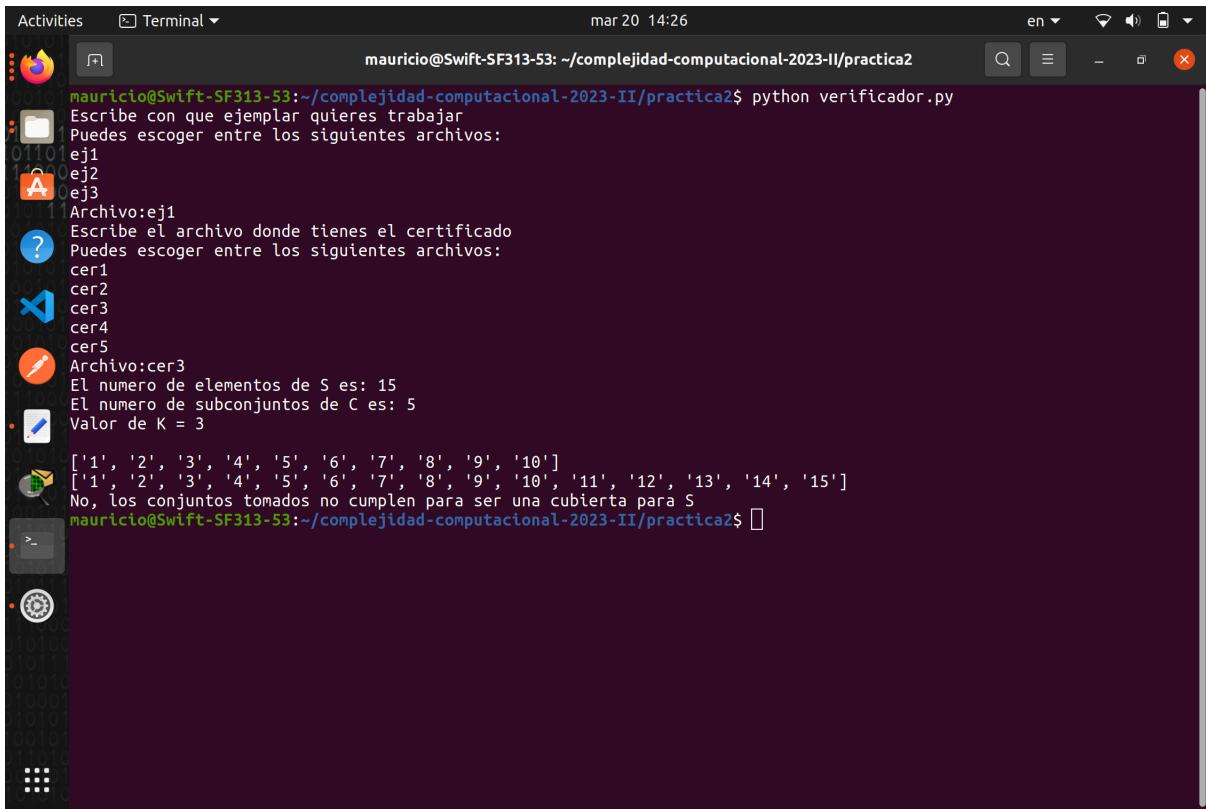
Figura 1: ejemplar 1 con el certificado 11010

```
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$ python verificador.py
Escribe con que ejemplar quieres trabajar
Puedes escoger entre los siguientes archivos:
ej1
ej2
ej3
Archivo:ej1
Escribe el archivo donde tienes el certificado
Puedes escoger entre los siguientes archivos:
cer1
cer2
cer3
cer4
cer5
Archivo:cer2
El numero de elementos de S es: 15
El numero de subconjuntos de C es: 5
Valor de K = 3
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15']
[['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15']]
Si, los conjuntos tomados cumplen para ser una cubierta para S
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$
```

Figura 2: ejemplar 1 Con el certificado 11100

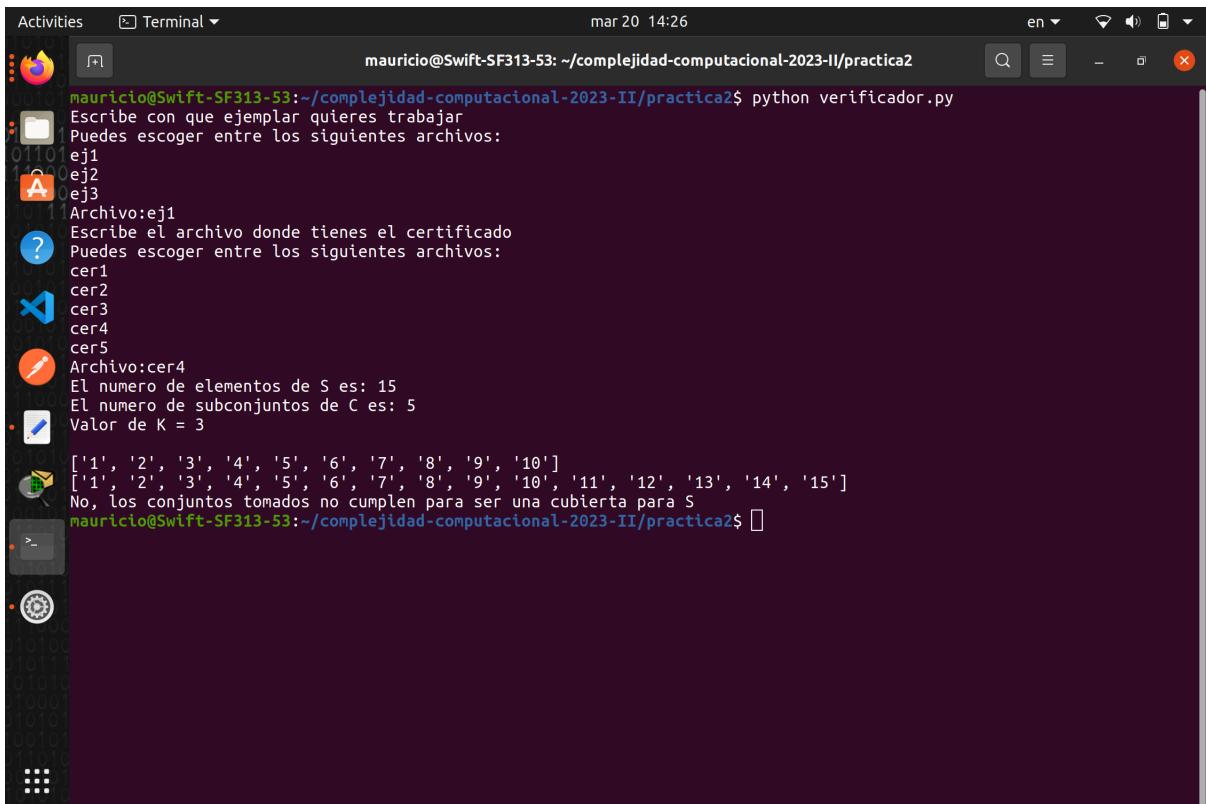
referencias

<https://mathworld.wolfram.com/Cover.html>



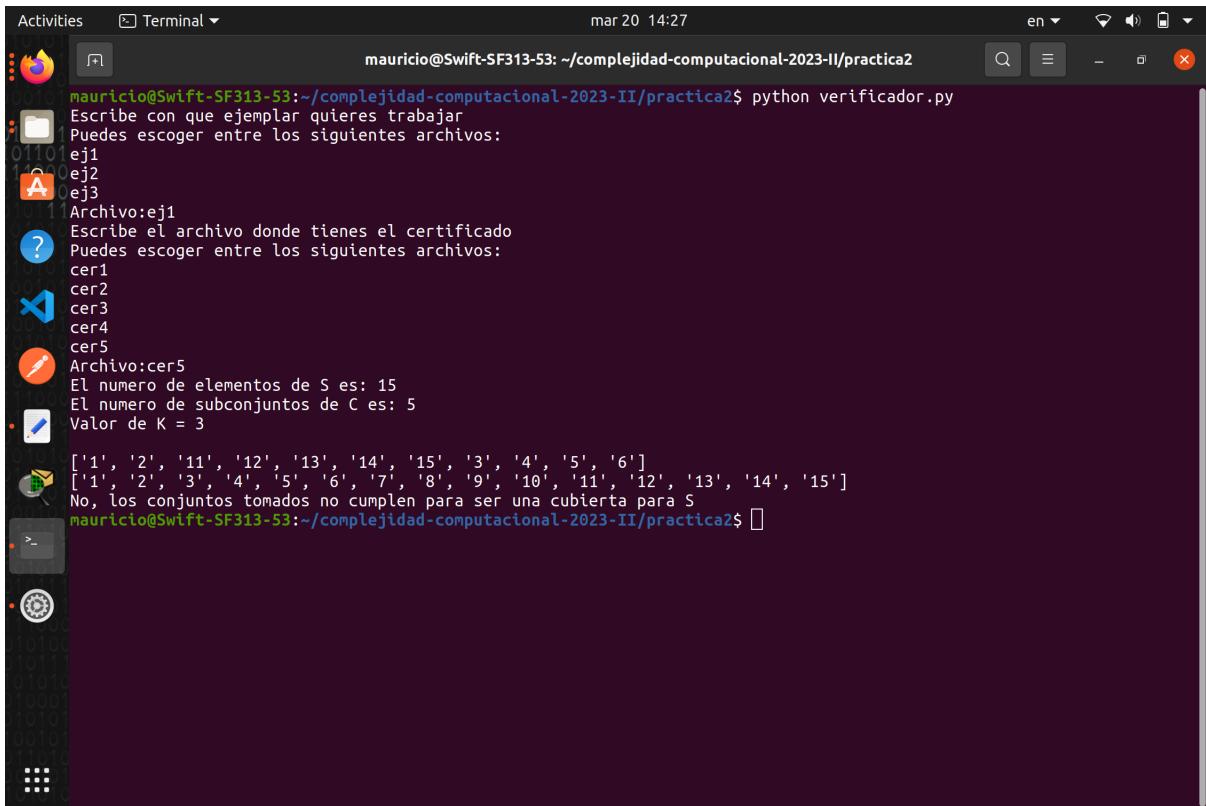
```
Activities Terminal mar 20 14:26
mauricio@Swift-SF313-53: ~/complejidad-computacional-2023-II/practica2$ python verificador.py
Escribe con que ejemplar quieres trabajar
Puedes escoger entre los siguientes archivos:
ej1
ej2
ej3
Archivo:ej1
Escribe el archivo donde tienes el certificado
Puedes escoger entre los siguientes archivos:
cer1
cer2
cer3
cer4
cer5
Archivo:cer3
El numero de elementos de S es: 15
El numero de subconjuntos de C es: 5
Valor de K = 3
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10']
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15']
No, los conjuntos tomados no cumplen para ser una cubierta para S
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$
```

Figura 3: ejemplar 1 Con el certificado 11001



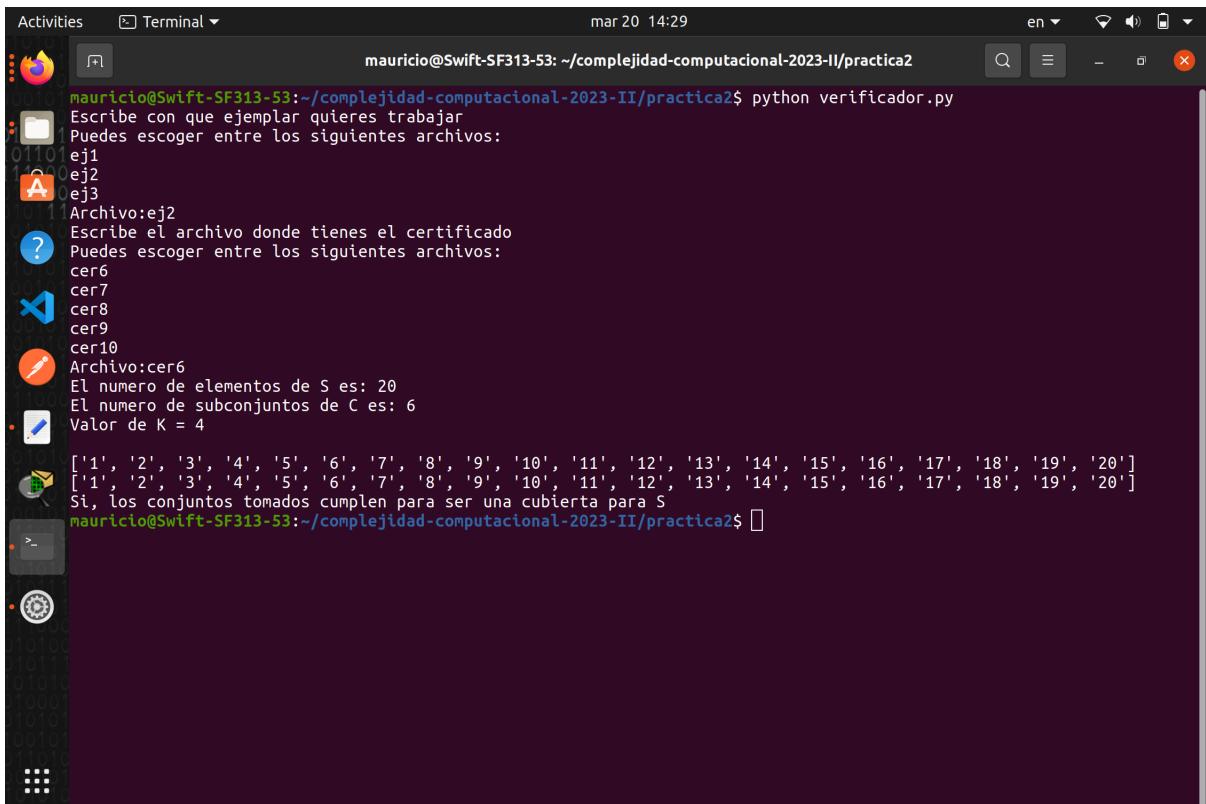
```
Activities Terminal mar 20 14:26
mauricio@Swift-SF313-53: ~/complejidad-computacional-2023-II/practica2$ python verificador.py
Escribe con que ejemplar quieres trabajar
Puedes escoger entre los siguientes archivos:
ej1
ej2
ej3
Archivo:ej1
Escribe el archivo donde tienes el certificado
Puedes escoger entre los siguientes archivos:
cer1
cer2
cer3
cer4
cer5
Archivo:cer4
El numero de elementos de S es: 15
El numero de subconjuntos de C es: 5
Valor de K = 3
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10']
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15']
No, los conjuntos tomados no cumplen para ser una cubierta para S
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$
```

Figura 4: ejemplar 1 Con el certificado 01011



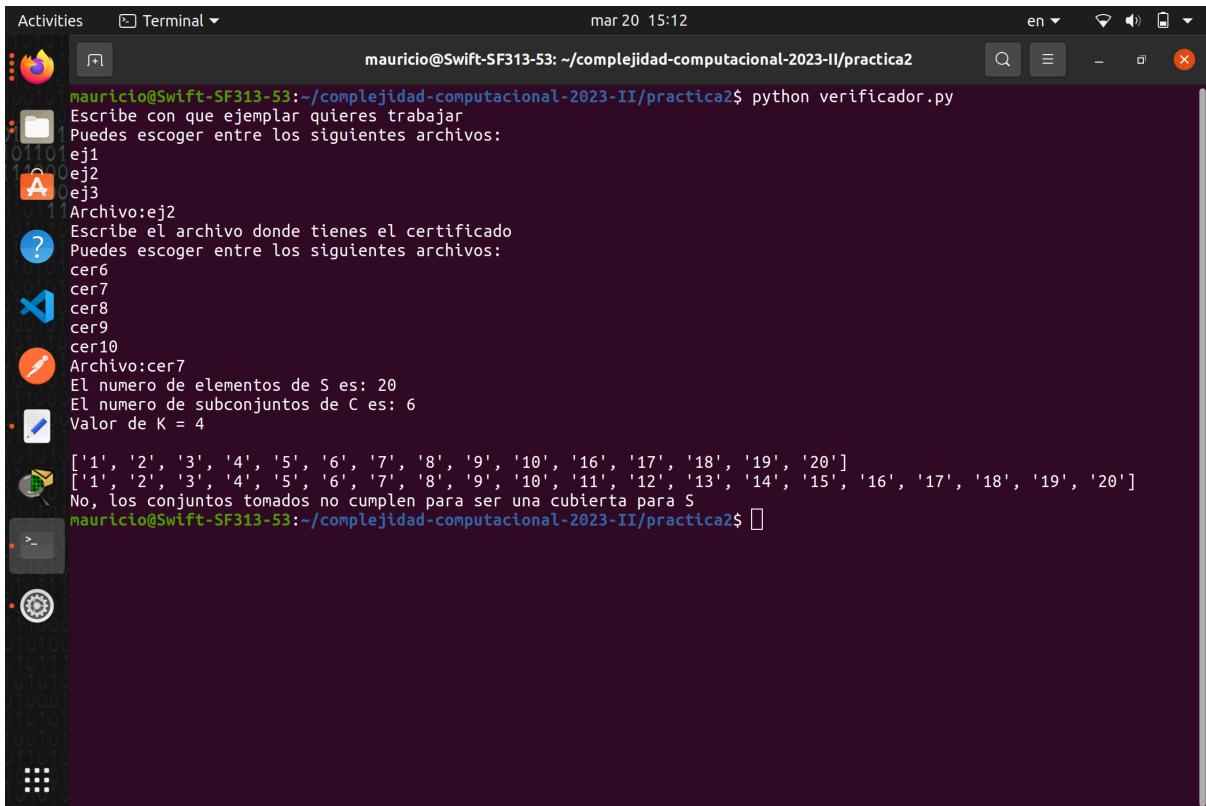
```
Activities Terminal mar 20 14:27
mauricio@Swift-SF313-53: ~/complejidad-computacional-2023-II/practica2$ python verificador.py
Escribe con que ejemplar quieres trabajar
Puedes escoger entre los siguientes archivos:
ej1
ej2
ej3
Archivo:ej1
Escribe el archivo donde tienes el certificado
Puedes escoger entre los siguientes archivos:
cer1
cer2
cer3
cer4
cer5
Archivo:cer5
El numero de elementos de S es: 15
El numero de subconjuntos de C es: 5
Valor de K = 3
['1', '2', '11', '12', '13', '14', '15', '3', '4', '5', '6']
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15']
No, los conjuntos tomados no cumplen para ser una cubierta para S
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$
```

Figura 5: ejemplar 1 Con el certificado 10110



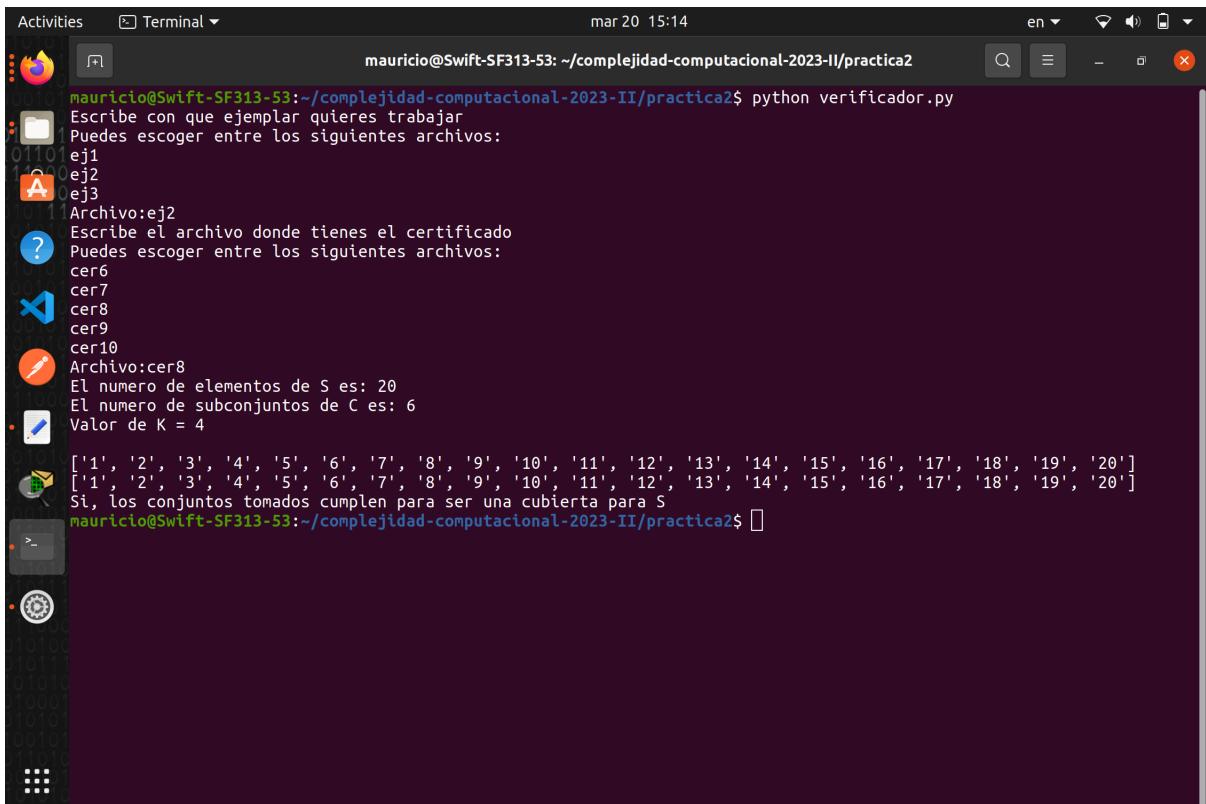
```
Activities Terminal mar 20 14:29
mauricio@Swift-SF313-53: ~/complejidad-computacional-2023-II/practica2$ python verificador.py
Escribe con que ejemplar quieres trabajar
Puedes escoger entre los siguientes archivos:
ej1
ej2
ej3
Archivo:ej2
Escribe el archivo donde tienes el certificado
Puedes escoger entre los siguientes archivos:
cer6
cer7
cer8
cer9
cer10
Archivo:cer6
El numero de elementos de S es: 20
El numero de subconjuntos de C es: 6
Valor de K = 4
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20']
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20']
Sí, los conjuntos tomados cumplen para ser una cubierta para S
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$
```

Figura 6: ejemplar 2 Con el certificado 111001



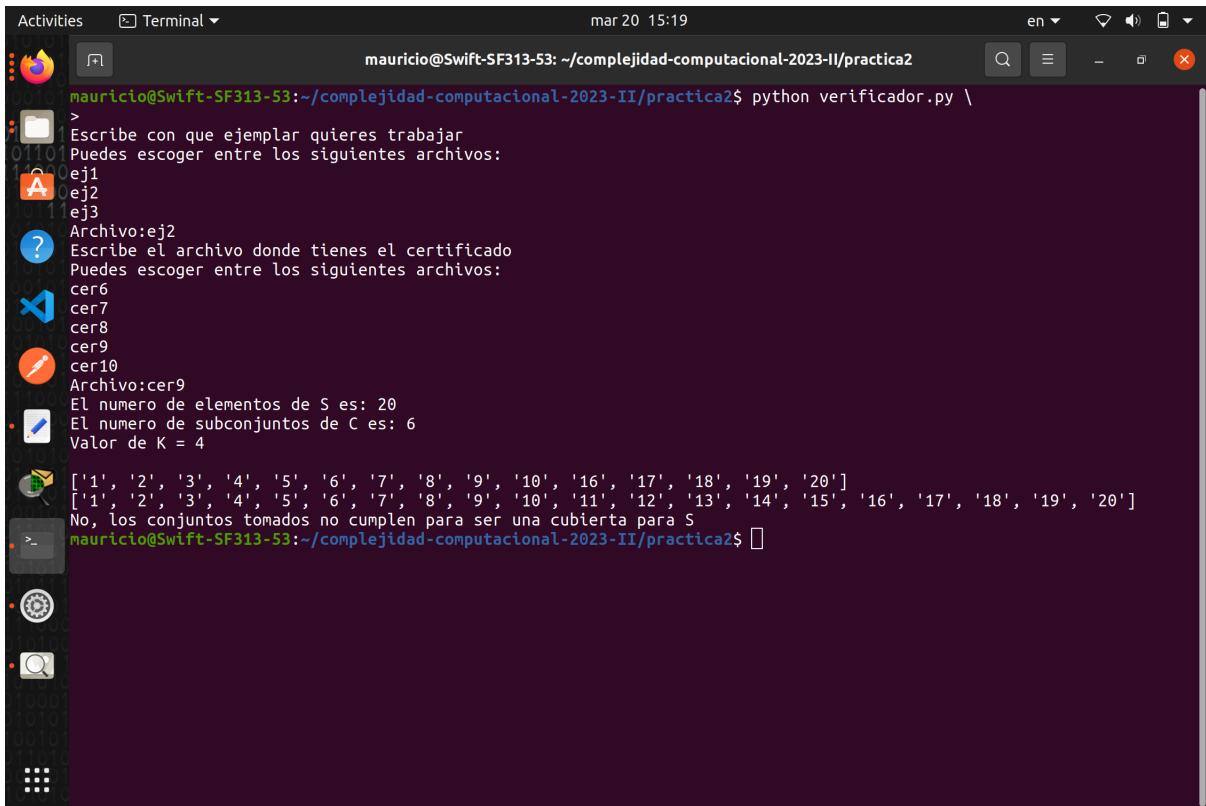
```
Activities Terminal mar 20 15:12
mauricio@Swift-SF313-53: ~/complejidad-computacional-2023-II/practica2$ python verificador.py
Escribe con que ejemplar quieres trabajar
Puedes escoger entre los siguientes archivos:
ej1
ej2
ej3
Archivo:ej2
Escribe el archivo donde tienes el certificado
Puedes escoger entre los siguientes archivos:
cer6
cer7
cer8
cer9
cer10
Archivo:cer7
El numero de elementos de S es: 20
El numero de subconjuntos de C es: 6
Valor de K = 4
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '16', '17', '18', '19', '20']
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20']
No, los conjuntos tomados no cumplen para ser una cubierta para S
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$
```

Figura 7: ejemplar 2 Con el certificado 110101



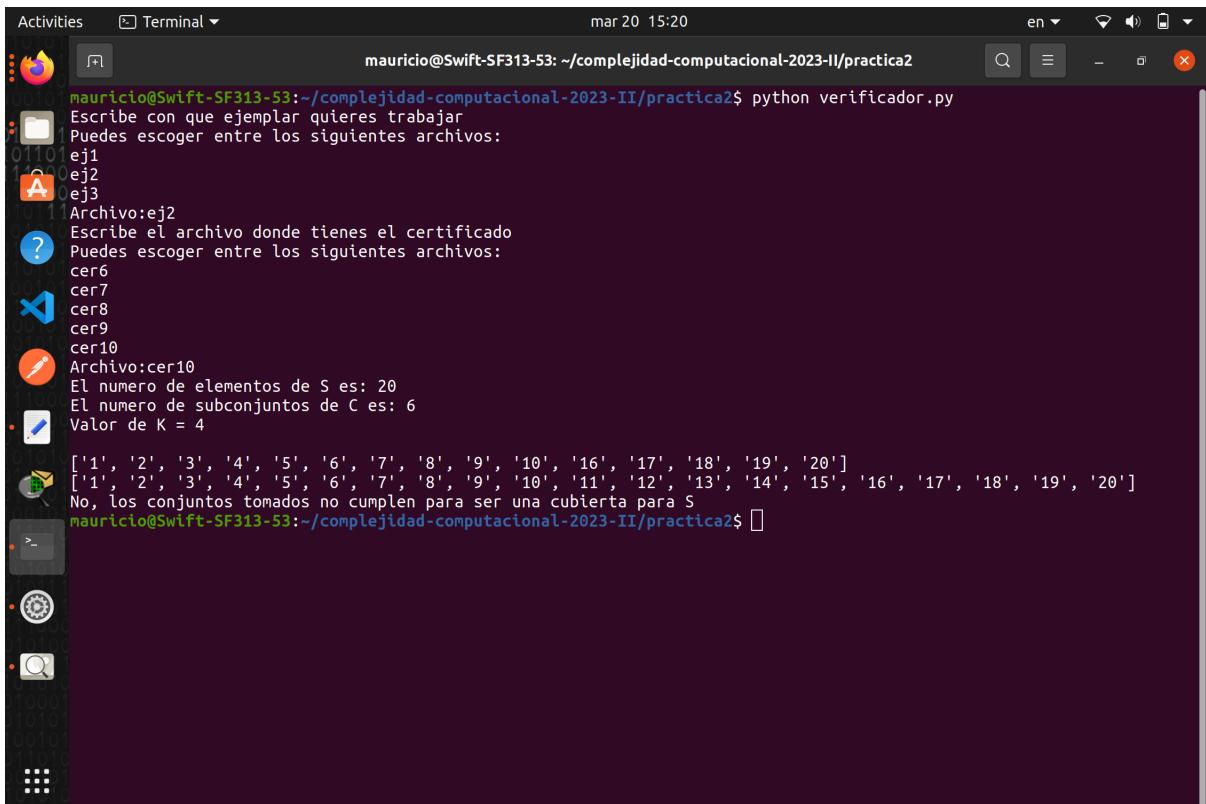
```
Activities Terminal mar 20 15:14
mauricio@Swift-SF313-53: ~/complejidad-computacional-2023-II/practica2$ python verificador.py
Escribe con que ejemplar quieres trabajar
Puedes escoger entre los siguientes archivos:
ej1
ej2
ej3
Archivo:ej2
Escribe el archivo donde tienes el certificado
Puedes escoger entre los siguientes archivos:
cer6
cer7
cer8
cer9
cer10
Archivo:cer8
El numero de elementos de S es: 20
El numero de subconjuntos de C es: 6
Valor de K = 4
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20']
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20']
Si, los conjuntos tomados cumplen para ser una cubierta para S
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$
```

Figura 8: ejemplar 2 Con el certificado 011101



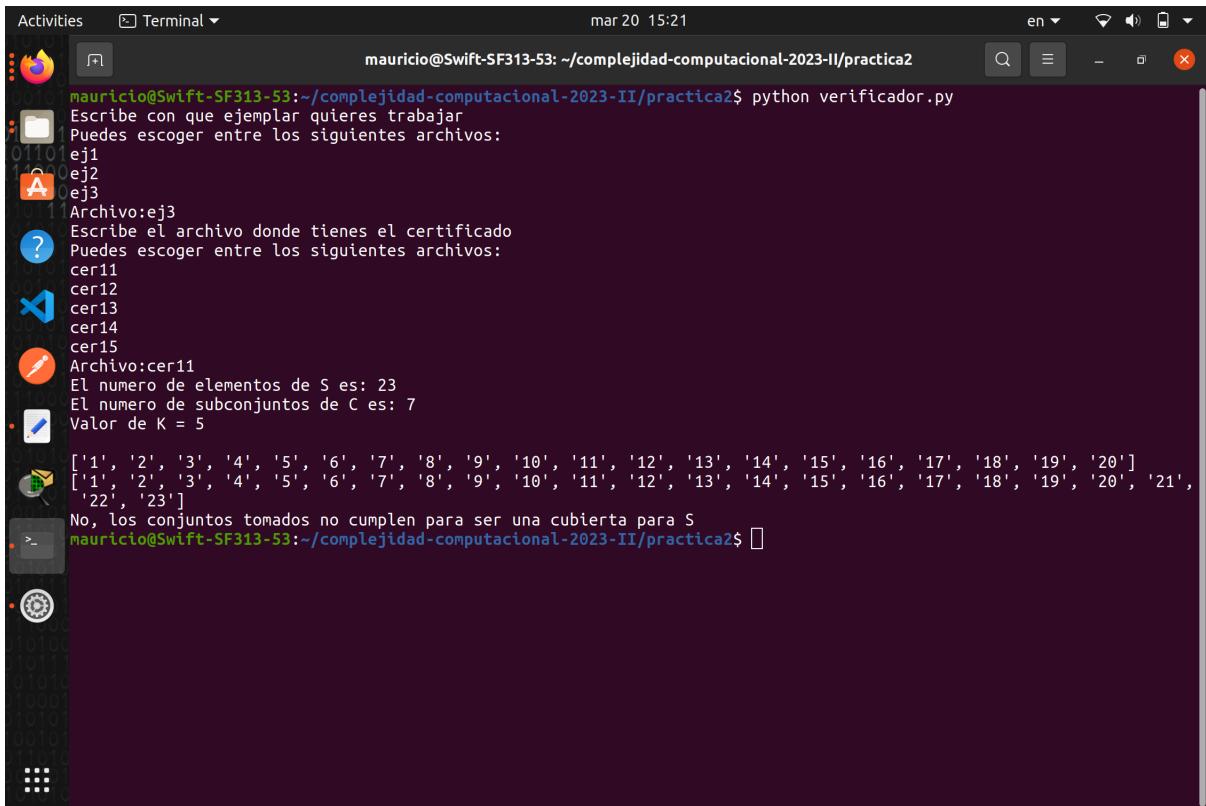
```
Activities Terminal mar 20 15:19 en
mauricio@Swift-SF313-53: ~/complejidad-computacional-2023-II/practica2$ python verificador.py \
>
Escribe con que ejemplar quieres trabajar
Puedes escoger entre los siguientes archivos:
ej1
ej2
ej3
Archivo:ej2
Escribe el archivo donde tienes el certificado
Puedes escoger entre los siguientes archivos:
cer6
cer7
cer8
cer9
cer10
Archivo:cer9
El numero de elementos de S es: 20
El numero de subconjuntos de C es: 6
Valor de K = 4
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '16', '17', '18', '19', '20']
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20']
No, los conjuntos tomados no cumplen para ser una cubierta para S
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$
```

Figura 9: ejemplar 2 Con el certificado 010111



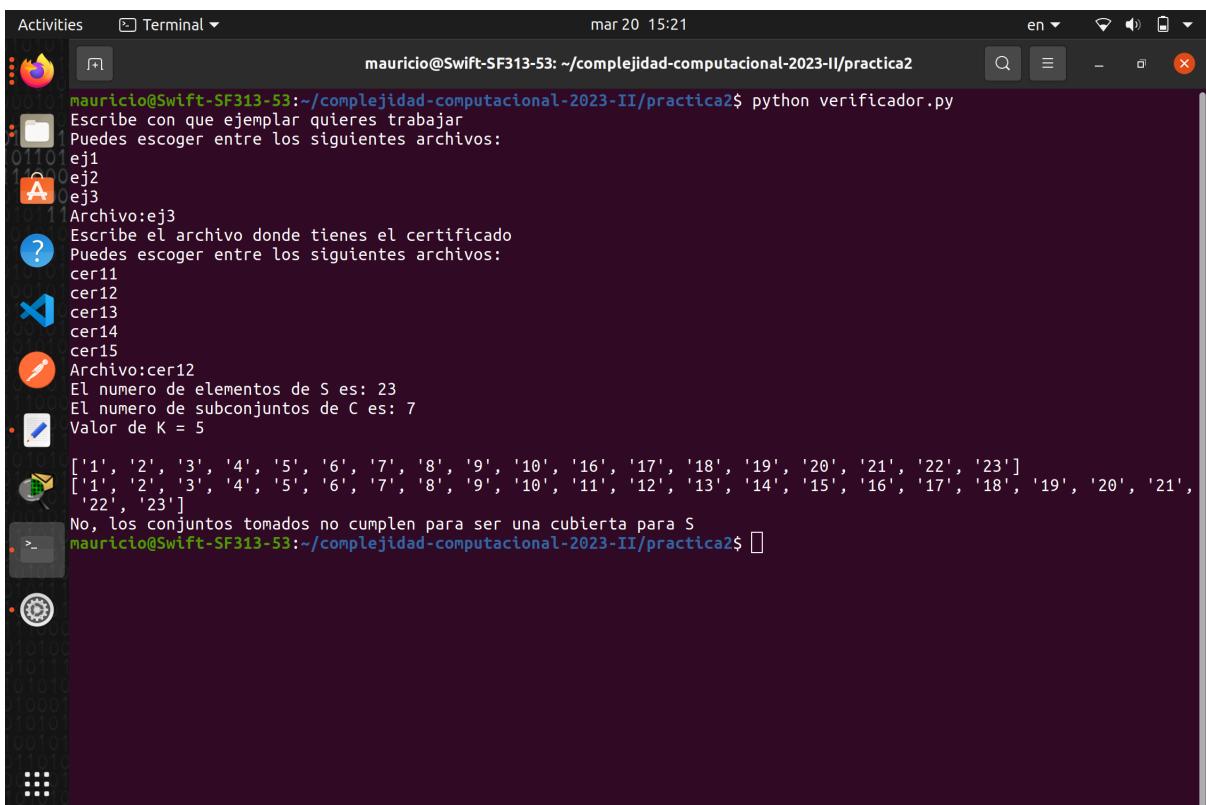
```
Activities Terminal mar 20 15:20 en
mauricio@Swift-SF313-53: ~/complejidad-computacional-2023-II/practica2$ python verificador.py \
>
Escribe con que ejemplar quieres trabajar
Puedes escoger entre los siguientes archivos:
ej1
ej2
ej3
Archivo:ej2
Escribe el archivo donde tienes el certificado
Puedes escoger entre los siguientes archivos:
cer6
cer7
cer8
cer9
cer10
Archivo:cer10
El numero de elementos de S es: 20
El numero de subconjuntos de C es: 6
Valor de K = 4
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '16', '17', '18', '19', '20']
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20']
No, los conjuntos tomados no cumplen para ser una cubierta para S
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$
```

Figura 10: ejemplar 2 Con el certificado 110011



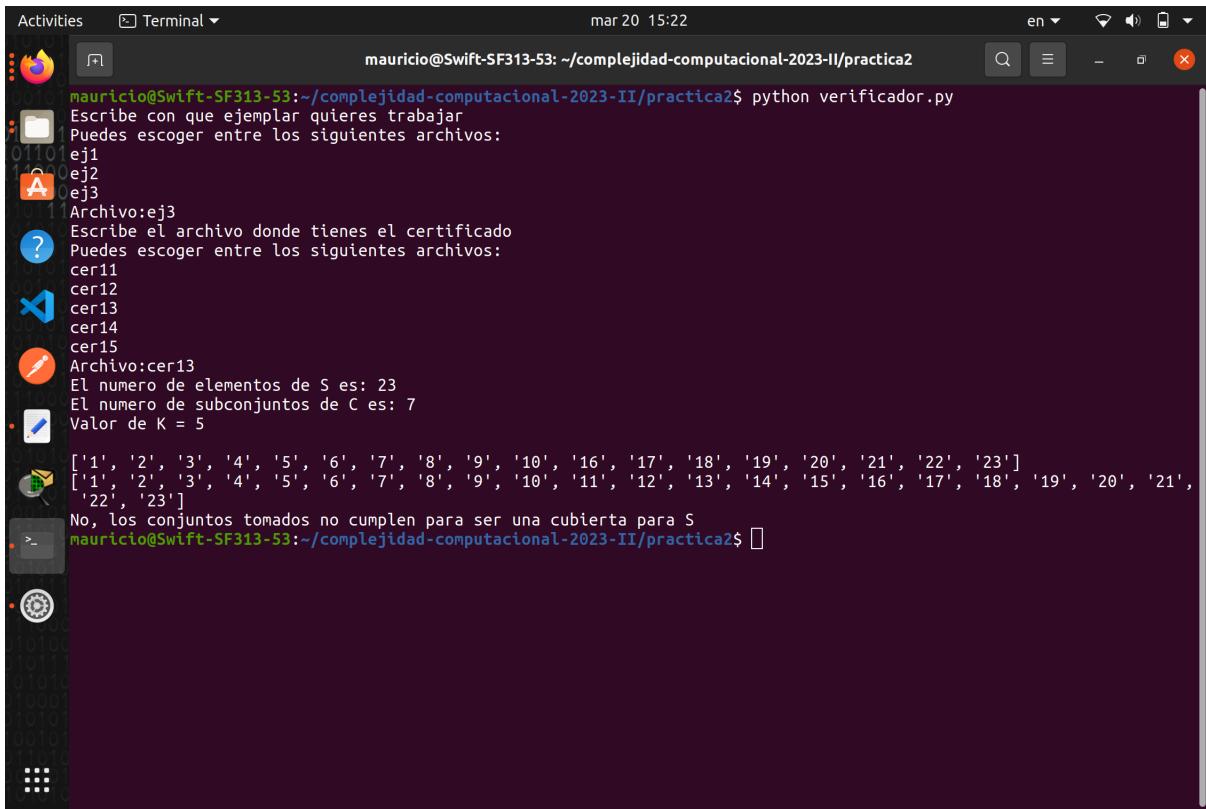
```
Activities Terminal mar 20 15:21 en ▾ mauricio@Swift-SF313-53: ~/complejidad-computacional-2023-II/practica2$ python verificador.py
Escribe con que ejemplar quieres trabajar
Puedes escoger entre los siguientes archivos:
ej1
ej2
ej3
Archivo:ej3
Escribe el archivo donde tienes el certificado
Puedes escoger entre los siguientes archivos:
cer11
cer12
cer13
cer14
cer15
Archivo:cer11
El numero de elementos de S es: 23
El numero de subconjuntos de C es: 7
Valor de K = 5
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20']
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20', '21',
 '22', '23']
No, los conjuntos tomados no cumplen para ser una cubierta para S
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$
```

Figura 11: ejemplar 3 Con el certificado 1111010



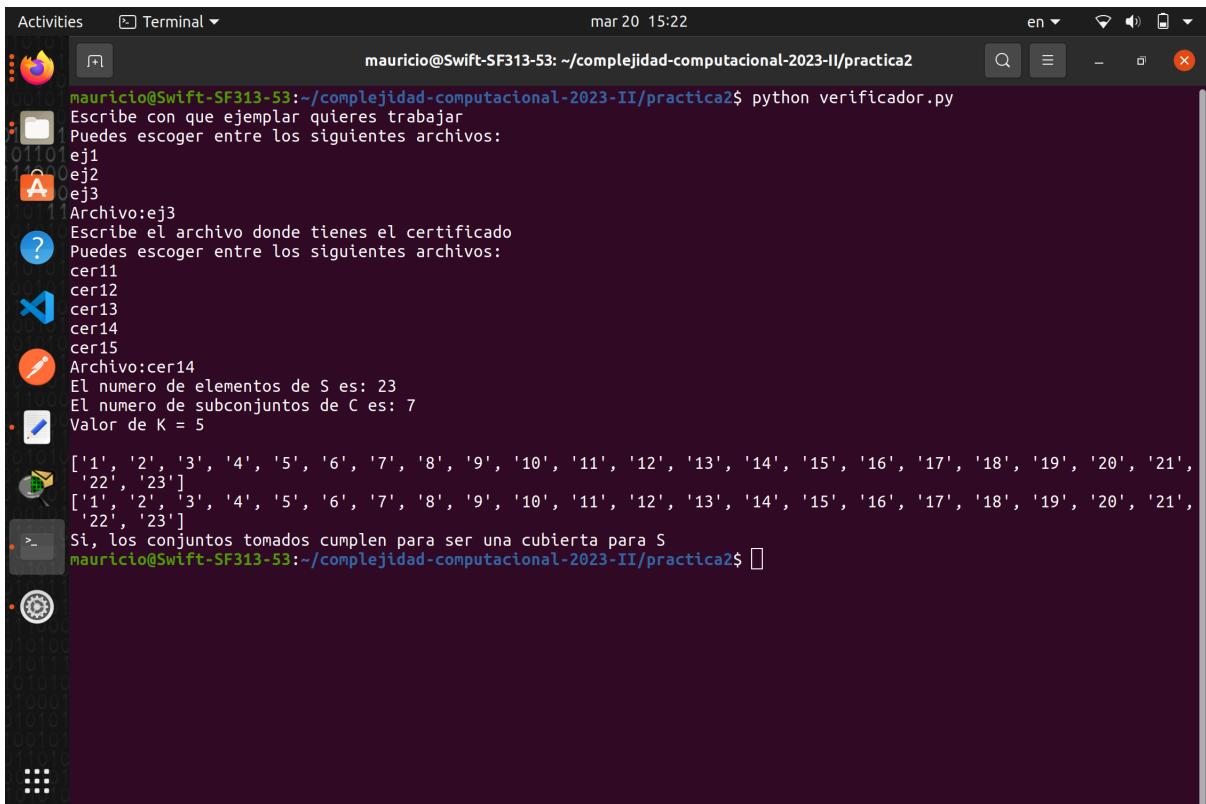
```
Activities Terminal mar 20 15:21 en ▾ mauricio@Swift-SF313-53: ~/complejidad-computacional-2023-II/practica2$ python verificador.py
Escribe con que ejemplar quieres trabajar
Puedes escoger entre los siguientes archivos:
ej1
ej2
ej3
Archivo:ej3
Escribe el archivo donde tienes el certificado
Puedes escoger entre los siguientes archivos:
cer11
cer12
cer13
cer14
cer15
Archivo:cer12
El numero de elementos de S es: 23
El numero de subconjuntos de C es: 7
Valor de K = 5
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '16', '17', '18', '19', '20', '21', '22', '23']
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20',
 '21', '22', '23']
No, los conjuntos tomados no cumplen para ser una cubierta para S
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$
```

Figura 12: ejemplar 3 Con el certificado 0101111



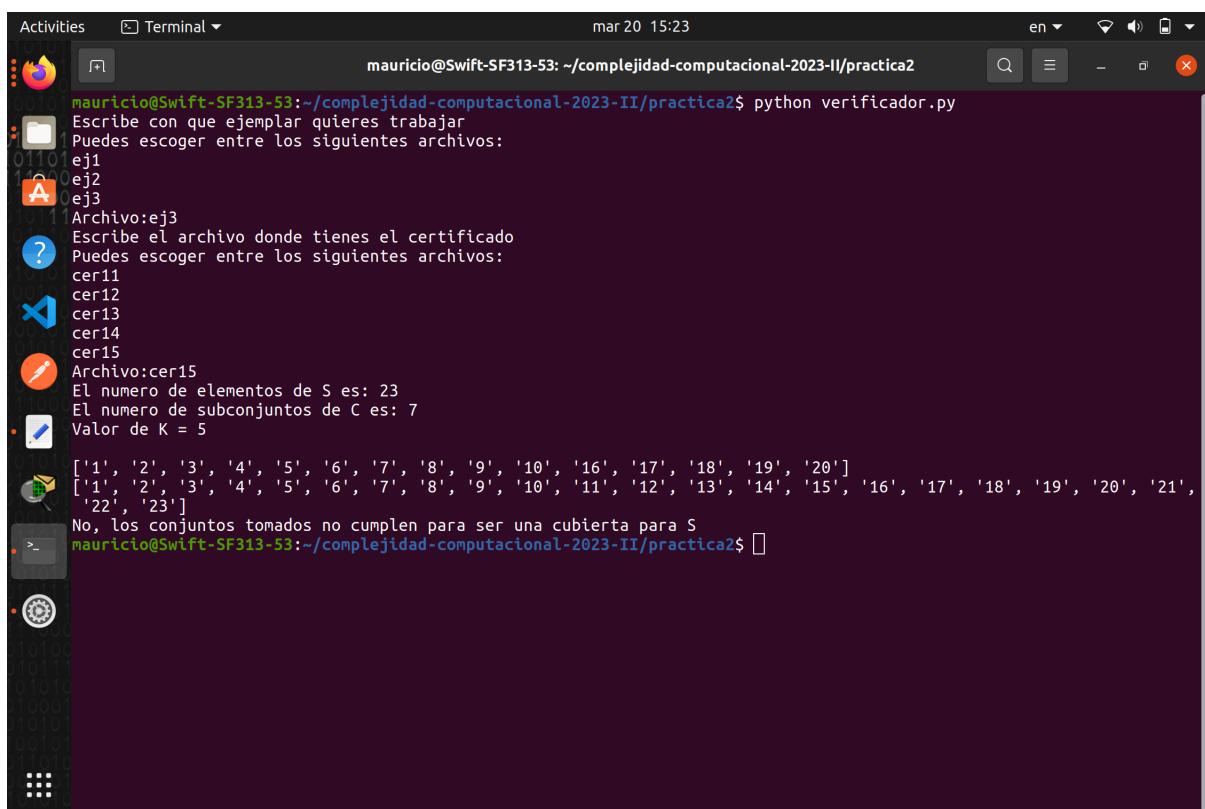
```
Activities Terminal mar 20 15:22 en ▾ mauricio@Swift-SF313-53: ~/complejidad-computacional-2023-II/practica2$ python verificador.py
Escribe con que ejemplar quieres trabajar
Puedes escoger entre los siguientes archivos:
ej1
ej2
ej3
Archivo:ej3
Escribe el archivo donde tienes el certificado
Puedes escoger entre los siguientes archivos:
cer11
cer12
cer13
cer14
cer15
Archivo:cer13
El numero de elementos de S es: 23
El numero de subconjuntos de C es: 7
Valor de K = 5
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '16', '17', '18', '19', '20', '21', '22', '23']
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20', '21',
 '22', '23']
No, los conjuntos tomados no cumplen para ser una cubierta para S
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$
```

Figura 13: ejemplar 3 Con el certificado 1100111



```
Activities Terminal mar 20 15:22 en ▾ mauricio@Swift-SF313-53: ~/complejidad-computacional-2023-II/practica2$ python verificador.py
Escribe con que ejemplar quieres trabajar
Puedes escoger entre los siguientes archivos:
ej1
ej2
ej3
Archivo:ej3
Escribe el archivo donde tienes el certificado
Puedes escoger entre los siguientes archivos:
cer11
cer12
cer13
cer14
cer15
Archivo:cer14
El numero de elementos de S es: 23
El numero de subconjuntos de C es: 7
Valor de K = 5
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20', '21',
 '22', '23']
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20', '21',
 '22', '23']
Si, los conjuntos tomados cumplen para ser una cubierta para S
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$
```

Figura 14: ejemplar 3 Con el certificado 0110111

A screenshot of a Linux desktop environment showing a terminal window. The terminal window has a dark background and contains the following text:

```
mauricio@Swift-SF313-53:~/complejidad-computacional-2023-II/practica2$ python verificador.py
Escribe con que ejemplar quieres trabajar
Puedes escoger entre los siguientes archivos:
ej1
ej2
A ej3
Archivo:ej3
Escribe el archivo donde tienes el certificado
Puedes escoger entre los siguientes archivos:
cer11
cer12
cer13
cer14
cer15
Archivo:cer15
El numero de elementos de S es: 23
El numero de subconjuntos de C es: 7
Valor de K = 5
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '16', '17', '18', '19', '20']
['1', '2', '3', '4', '5', '6', '7', '8', '9', '10', '11', '12', '13', '14', '15', '16', '17', '18', '19', '20', '21',
'22', '23']
No, los conjuntos tomados no cumplen para ser una cubierta para S
```

Figura 15: ejemplar 3 Con el certificado 1101110