a

February 8, 2025

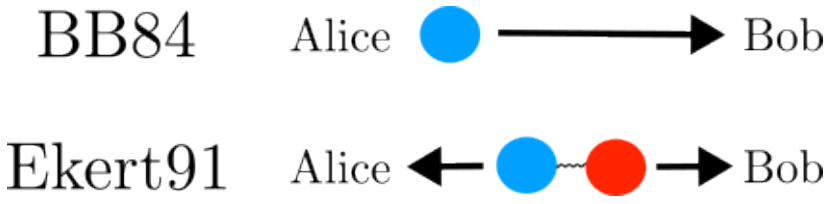
[ ]:
```
pip install -U --force --no-deps git+https://github.com/MIROptics/ECC2025.git
```

[2]:
```
import numpy as np
from qiskit import QuantumCircuit, QuantumRegister
from qiskit_aer import AerSimulator
from ECC2025.testing import test_4a, test_4b, test_4c
```

1 Criptografía Cuántica

La criptografía cuántica es una rama de la información cuántica enfocada en el desarrollo y análisis de protocolos de comunicación seguros, cuya seguridad se fundamenta en las leyes fundamentales de la mecánica cuántica. Uno de los primeros y más sencillos ejemplos de estos protocolos es el BB84, desarrollado por Bennett y Brassard y estudiado en la escuela. Este protocolo se basa en el envío de estados cuánticos aleatorios de Alice a Bob, quien los mide para generar una clave compartida. Una característica destacable del BB84 es que no requiere el uso de estados entrelazados.

En este desafío exploraremos el protocolo EKERT91, que se diferencia del BB84 al emplear estados entrelazados. Esta característica permite detectar posibles espías mediante la verificación de las desigualdades de Bell, basándose en los resultados de las mediciones realizadas por Alice y Bob. Una de las principales ventajas del uso de entrelazamiento en la distribución de la llave es que el protocolo ofrece mayor robustez frente a ciertos tipos de ataques. La intervención de un espía rompería el entrelazamiento, lo que garantiza la seguridad del protocolo al apoyarse en el teorema de Bell.



Antes de procede con el desafío revisaremos el esquema general del protocolo EKERT91

1.0.1 1. Se generan pares entrelazados

- Una fuente genera pares de partículas en un estado de singlete, que es un estado cuántico maximalmente entrelazado.

- Alice y Bob reciben cada uno una partícula de cada par generado.

1.0.2 2. Se acuerdan las bases de medición

- Alice y Bob acuerdan previamente un conjunto de bases sobre las cuales medirán las partículas.

- En particular, cada uno dispone de tres posibles bases de medición, de las cuales dos coinciden entre ambos. Estas bases compartidas permiten obtener resultados correlacionados.

- La elección de la base para cada medición se realiza de forma aleatoria en cada caso.

1.0.3 3. Se realizan las mediciones

- Alice y Bob realizan sus mediciones sobre las partículas que reciben, utilizando las bases seleccionadas aleatoriamente.

1.0.4 4. Comparamos las bases que Alice y Bob midieron

- Una vez realizadas las mediciones, Alice y Bob anuncian públicamente cuáles fueron las bases que utilizaron para cada medición.

- Luego, separan los resultados en dos grupos:
  - **Grupo 1:** Mediciones donde las bases no coinciden.

  - **Grupo 2:** Mediciones donde las bases coinciden.

1.0.5 5. Evaluamos desigualdades de Bell y verificamos la seguridad

- Los resultados de las mediciones del **Grupo 1** se anuncian públicamente y se utilizan para evaluar la desigualdad de Bell.

- Si la desigualdad de Bell se viola, se garantiza que no hubo intervención de un espía, confirmando la seguridad del canal cuántico.

1.0.6 6. Generamos la clave compartida

- Una vez verificada la ausencia de espías, los resultados del **Grupo 2** se utilizan para generar la clave compartida.

- Es importante destacar que Alice puede deducir con precisión los resultados obtenidos por Bob, y viceversa, basándose únicamente en sus propias mediciones, ya que en este grupo los resultados están correlacionados debido al entrelazamiento cuántico.

Desafío: El primer paso es que Alice y Bob generen una lista de trits (0, 1, o 2) aleatoria. Estas serán las etiquetas de las bases sobre las que ellos mediran.

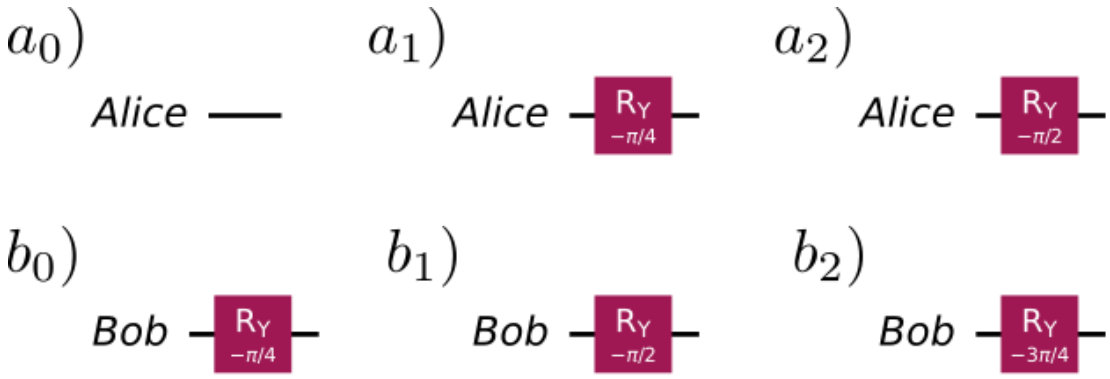
[3]:
```
num_trits = 10000
alice_random_trits = np.random.randint(0,3,size=num_trits)
bob_random_trits = np.random.randint(0,3,size=num_trits)
print('Alice trits =', alice_random_trits )
print('')
```

```
print('Bob trits =', bob_random_trits )
```

```
Alice trits = [0 1 1 … 1 2 1]

Bob trits = [2 2 1 … 1 0 1]
```

Estos trits serán usados para crear circuitos cuánticos aleatorios combinando los siguientes circuitos para Alice y Bob



Desafío: Complete los circuitos de la siguiente celda para que implementen todos los circuitos anteriores. Las listas `Aj` y `Bj` contienen los circuitos de Alice y Bob, respectivamente.

[4]:
```
qa = QuantumRegister(1, name='Alice')
qb = QuantumRegister(1, name='Bob')

qc_a0 = QuantumCircuit(qa)
##### Escriba su solución acá

############

qc_a1 = QuantumCircuit(qa)
##### Escriba su solución acá
qc_a1.ry(-np.pi/4, 0)

############

qc_a2 = QuantumCircuit(qa)
##### Escriba su solución acá
qc_a2.ry(-np.pi/2, 0)

############

qc_b0 = QuantumCircuit(qb)
##### Escriba su solución acá
qc_b0.ry(-np.pi/4, 0)
```

```
############

qc_b1 = QuantumCircuit(qb)
##### Escriba su solución acá
qc_b1.ry(-np.pi/2, 0)

############

qc_b2 = QuantumCircuit(qb)
##### Escriba su solución acá
qc_b2.ry(-3*np.pi/4, 0)

############

Aj = [qc_a0,qc_a1,qc_a2]
Bk = [qc_b0,qc_b1,qc_b2]
```

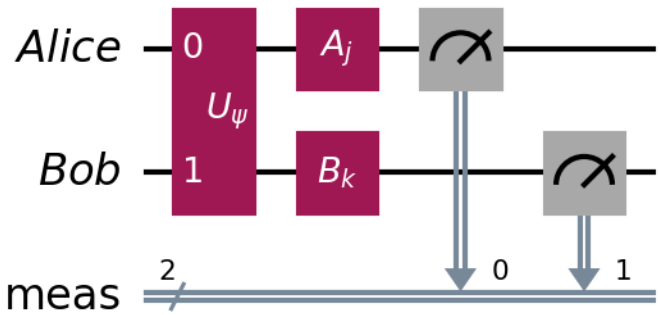
[5]:
```
test_4a( Aj, Bk )
```

```
Felicitaciones, tu solución es correcta!
```

El Ekert91 emplea un estado maximalmente estrelazado llamado singlete

$$|\psi\rangle = \frac{1}{\sqrt{2}}\left(|01\rangle - |10\rangle\right). \tag{1}$$

Este estado toma el rol de canal cuántico para la comunicación, enviando uno de los qubits a Alice y el otro a Bob. Posteriormente, Alice y Bob utilizan sus trits aleatorios y aplican alguno de los circuitos anteriores a su correspondiente qubit. El circuito completo tiene la siguiente forma:



Acá $U_\psi$es una operación unitaria que prepara el estado singlete, es decir$|\psi\rangle = U_\psi|00\rangle$, mientras que A_jy$B_k$son los circuitos contenidos en las listas `Aj` y `Bk`, con$j, k \in \{0, 1, 2\}$.

Desafío: Construya estos circuitos para cada par de trits de Alice y Bob.

[6]:
```
qcs = []
for i in range(num_trits):
    qc = QuantumCircuit( qa, qb )

    j = alice_random_trits[i]
    k = bob_random_trits[i]

    ##### Escriba su solución acá
    #qc.x(qa)
    #qc.x(qb)
    #qc.h(qa)
    #qc.cx(qa, qb)

    qc.h(0)
    qc.cx(0,1)
    qc.x(0)
    qc.z(1)

    qc.compose(Aj[j], qa, inplace=True)
    qc.compose(Bk[k], qb, inplace=True)
    #############

    qc.measure_all()
    qcs.append(qc)
```

[7]:
```
test_4b( qcs, alice_random_trits, bob_random_trits ) #takes a while
```

```
Felicitaciones, tu solución es correcta!
```

Simulando los experiementos de cada uno de estos circuitos.

[8]:
```
simulator = AerSimulator()

job = simulator.run( qcs, shots=1 )
counts_ekert = job.result().get_counts()
```

Despues de las medidas, Alice y Bob hacen publicos sus listas de trits y separan sus medidas en dos grupos. El primer grupo consiste en aquellos con trits $(a, b) \in \{(0, 0), (0, 2), (2, 0), (2, 2)\}$, que corresponde al grupo donde las bases medidas difieren. Estas mediciones nos permiten verificar si hay algún espía en la comunicación gracias al teorema de Bell. Para esto debemos evaluar la siguiente cantidad:

$$S = E_{00} - E_{02} + E_{20} + E_{22}, \tag{2}$$

donde

$$E_{jk} = p(00|jk) + p(11|jk) - p(01|jk) - p(10|jk), \tag{3}$$

y $p(lm|jk)$es la probabilidad de obtener el resultadolmal medir el circuitojk.

Esta cantidad debe tener un valor $|S| \approx 2\sqrt{2}$, lo cual representa que el estado esta maximalmente entrelazado. En el caso que $|S| < 2\sqrt{2}$se tiene que el canal ha perdido entrelazamiento, la cual se puede deber, entre otras cosas, a la presencia de un espía en la comunicación. Si$|S|$es menor a 2, es decir$|S| < 2$, el canal cuántico perdió completamente su entrelazamiento y su seguridad.

La siguiente celda muestra como calcula S usando los resultados del primer grupo.

[9]:
```
bell = 0

ExpVal = np.zeros([3,3])
times_per_ExpVal = np.zeros([3,3])

for j in range(num_trits):

    a = alice_random_trits[j]
    b = bob_random_trits[j]

    E = counts_ekert[j].get('00',0) + counts_ekert[j].get('11',0) -␣
 ↪counts_ekert[j].get('10',0) - counts_ekert[j].get('01',0)

    ExpVal[a,b] += E
    times_per_ExpVal[a,b] += 1

ExpVal = ExpVal / times_per_ExpVal

S = ExpVal[ 0,0 ] - ExpVal[ 0,2] + ExpVal[ 2,0 ] + ExpVal[ 2,2]

print( S )
```

```
-2.9405393314323094
```

El segundo grupo son aquellos con trits $(a, b) \in \{(1, 0), (2, 1)\}$, que corresponde al grupo donde las bases medidas coinciden. Las mediciones de estos circuitos estarán anticorrelacionadas debido al estado singlete, es decir, si Alice mide 0, Bob medirá 1, y viceversa. Esta estructura nos permite establecer una llave compartida entre Alice y Bob.

Desafío: Contruya la llave secreta usando los resultados de las mediciones de Alice. Cada bit de la clave debe ser un elemento de la lista `key`.

Pista: Recuerde que los resultados de las medidas están en `counts_ekert`, usted debe pensar como extraer a información requerida de esa variable usando herramientas de python

[ ]:
```
key = []

##### Escriba su solución acá

count = 0
for j in range(num_trits):
    a = alice_random_trits[j]
    b = bob_random_trits[j]
```

```
    if (a == 1 and b == 0) or (a == 2 and b == 1):
      key.append(list(counts_ekert[j].keys())[0][0])

############
print( key )
```

```
['0', '0', '0', '0', '1', '0', '0', '0', '0', '1', '1', '1', '1', '0', '1', '1',
'1', '1', '0', '0', '0', '1', '0', '1', '1', '0', '1', '1', '0', '1', '0', '1',
'1', '1', '1', '1', '1', '1', '1', '0', '0', '1', '1', '1', '0', '0', '1', '0',
'1', '0', '0', '1', '1', '0', '1', '1', '0', '0', '0', '0', '1', '0', '1', '1',
'1', '0', '0', '0', '1', '0', '0', '1', '0', '1', '0', '1', '1', '0', '0', '0',
'0', '0', '0', '1', '1', '0', '1', '1', '1', '0', '0', '1', '1', '1', '0', '1',
'0', '0', '0', '1', '0', '1', '1', '0', '0', '1', '0', '1', '0', '0', '0', '0',
'1', '0', '1', '1', '0', '0', '0', '0', '1', '0', '0', '0', '0', '1', '1', '1',
'1', '1', '0', '1', '0', '1', '0', '1', '1', '1', '1', '1', '0', '0', '1', '1',
'1', '1', '0', '0', '1', '0', '0', '1', '0', '0', '0', '0', '1', '1', '1', '1',
'0', '0', '1', '1', '0', '1', '1', '0', '1', '0', '1', '1', '0', '1', '1', '0',
'0', '1', '1', '1', '0', '1', '0', '0', '1', '1', '0', '0', '0', '0', '1', '0',
'1', '1', '0', '0', '0', '1', '0', '1', '0', '1', '0', '0', '0', '0', '1', '0',
'0', '0', '1', '0', '0', '1', '0', '0', '0', '1', '0', '1', '0', '0', '1', '0',
'0', '0', '1', '0', '0', '0', '1', '0', '1', '0', '0', '0', '0', '1', '0', '1',
'1', '1', '0', '1', '1', '1', '1', '1', '0', '1', '0', '0', '0', '1', '0', '1',
'1', '0', '1', '0', '0', '1', '1', '0', '1', '0', '0', '1', '0', '0', '0', '0',
'1', '0', '1', '1', '0', '0', '1', '1', '0', '0', '1', '1', '1', '0', '0', '1',
'1', '1', '0', '0', '1', '1', '1', '1', '1', '1', '0', '1', '1', '1', '1', '0',
'0', '1', '1', '1', '1', '1', '0', '1', '1', '0', '0', '1', '1', '1', '0', '0',
'0', '1', '0', '0', '1', '0', '0', '0', '1', '1', '0', '1', '0', '0', '0', '1',
'1', '0', '1', '0', '1', '0', '0', '0', '0', '1', '1', '1', '1', '0', '0', '0',
'1', '1', '1', '0', '1', '0', '0', '1', '1', '0', '1', '1', '0', '1', '1', '1',
'0', '1', '1', '0', '1', '1', '0', '1', '0', '0', '0', '0', '0', '0', '1', '0',
'1', '1', '1', '1', '0', '1', '0', '0', '1', '0', '1', '1', '0', '0', '0', '0',
'1', '1', '0', '1', '1', '0', '0', '1', '1', '0', '0', '1', '1', '0', '1', '0',
'0', '1', '1', '1', '0', '1', '0', '1', '0', '1', '1', '1', '1', '1', '1', '0',
'0', '0', '1', '0', '1', '0', '1', '1', '0', '0', '0', '0', '0', '0', '0', '1',
'0', '0', '1', '1', '0', '0', '1', '0', '0', '0', '1', '1', '1', '0', '1', '1',
'1', '0', '0', '1', '0', '0', '0', '1', '1', '0', '0', '1', '0', '1', '1', '0',
'0', '1', '1', '0', '1', '0', '0', '1', '1', '1', '0', '1', '0', '1', '1', '1',
'1', '1', '0', '0', '1', '0', '1', '1', '0', '1', '0', '1', '1', '1', '0', '1',
'1', '1', '0', '0', '1', '0', '1', '1', '0', '1', '0', '0', '0', '0', '0', '0',
'0', '0', '0', '1', '1', '1', '0', '1', '0', '1', '1', '0', '0', '0', '1', '1',
'0', '0', '0', '1', '1', '1', '0', '1', '0', '1', '1', '1', '1', '1', '0', '0',
'1', '1', '1', '1', '0', '0', '1', '0', '0', '0', '0', '1', '0', '1', '1', '0',
'0', '1', '0', '0', '0', '0', '0', '1', '1', '0', '0', '1', '1', '0', '1', '0',
'1', '1', '1', '0', '1', '1', '0', '1', '0', '1', '1', '0', '1', '0', '0', '1',
'0', '1', '1', '0', '1', '1', '1', '1', '0', '0', '1', '0', '0', '0', '0', '1',
'0', '1', '1', '1', '0', '0', '1', '0', '1', '1', '0', '0', '0', '0', '0', '1',
'1', '0', '0', '0', '0', '0', '1', '0', '0', '1', '1', '0', '1', '1', '0', '0',
```

'1', '0', '0', '0', '1', '0', '1', '1', '0', '1', '0', '0', '0', '1', '0', '0',
'0', '0', '0', '1', '0', '1', '0', '0', '1', '1', '0', '0', '1', '0', '0', '0',
'1', '1', '0', '1', '0', '0', '1', '0', '1', '0', '1', '1', '1', '0', '0', '0',
'0', '1', '0', '0', '1', '1', '1', '0', '1', '1', '0', '1', '0', '1', '0', '1',
'0', '1', '1', '1', '1', '1', '1', '0', '0', '1', '0', '0', '1', '1', '0', '1',
'1', '1', '1', '0', '0', '0', '1', '1', '0', '1', '0', '0', '1', '0', '0', '1',
'1', '1', '1', '0', '1', '0', '1', '0', '0', '1', '0', '1', '0', '1', '0', '1',
'0', '0', '1', '0', '1', '0', '1', '1', '1', '1', '0', '1', '0', '0', '0', '1',
'1', '1', '0', '0', '1', '0', '1', '0', '1', '1', '0', '1', '1', '0', '0', '0',
'1', '0', '1', '1', '1', '1', '0', '0', '0', '0', '0', '0', '0', '0', '0', '0',
'1', '1', '1', '0', '1', '0', '0', '0', '1', '1', '0', '0', '1', '1', '1', '0',
'0', '1', '0', '0', '1', '1', '0', '0', '1', '0', '0', '0', '0', '1', '0', '1',
'1', '1', '0', '1', '1', '0', '1', '0', '0', '1', '1', '1', '0', '0', '0', '0',
'0', '0', '1', '1', '0', '1', '0', '0', '1', '1', '1', '0', '0', '0', '1', '0',
'0', '1', '1', '1', '0', '1', '0', '1', '1', '1', '1', '1', '0', '1', '1', '0',
'1', '0', '1', '0', '1', '0', '1', '1', '1', '1', '0', '0', '0', '0', '0', '1',
'1', '0', '0', '1', '0', '1', '1', '1', '1', '1', '0', '1', '0', '0', '1', '0',
'0', '1', '0', '1', '1', '1', '1', '1', '1', '0', '0', '1', '0', '1', '0', '1',
'0', '0', '0', '1', '0', '0', '0', '1', '0', '0', '0', '1', '0', '1', '1', '0',
'0', '1', '0', '1', '1', '0', '0', '1', '0', '1', '0', '0', '0', '0', '0', '0',
'0', '1', '0', '0', '1', '0', '1', '0', '1', '1', '0', '1', '1', '0', '0', '0',
'1', '1', '1', '0', '1', '1', '0', '0', '0', '1', '0', '1', '0', '0', '1', '0',
'1', '0', '0', '0', '1', '0', '0', '1', '0', '1', '1', '1', '1', '0', '1', '1',
'0', '1', '1', '1', '0', '1', '0', '0', '0', '0', '1', '0', '1', '0', '1', '0',
'1', '1', '0', '0', '0', '1', '1', '0', '0', '0', '0', '1', '1', '0', '0', '0',
'1', '1', '0', '1', '0', '0', '1', '0', '1', '0', '0', '1', '1', '1', '1', '1',
'0', '0', '1', '0', '1', '0', '0', '0', '1', '0', '0', '1', '1', '0', '0', '0',
'1', '1', '0', '0', '1', '1', '0', '1', '1', '0', '1', '1', '1', '0', '0', '1',
'0', '1', '0', '1', '1', '1', '1', '0', '0', '1', '0', '0', '0', '0', '0', '1',
'1', '1', '0', '1', '1', '0', '1', '1', '1', '1', '1', '1', '1', '0', '1', '0',
'1', '0', '0', '1', '0', '0', '0', '0', '1', '1', '0', '1', '0', '1', '0', '0',
'1', '1', '1', '1', '0', '0', '0', '0', '1', '0', '1', '1', '0', '1', '1', '1',
'1', '0', '1', '1', '1', '0', '0', '1', '1', '1', '1', '1', '0', '1', '1', '1',
'0', '1', '0', '0', '1', '0', '0', '0', '1', '1', '1', '1', '0', '0', '1', '0',
'0', '1', '1', '1', '0', '0', '0', '1', '1', '0', '1', '0', '1', '1', '0', '0',
'0', '0', '0', '0', '1', '1', '1', '1', '0', '1', '1', '0', '0', '1', '1', '1',
'1', '1', '0', '1', '1', '1', '1', '1', '1', '0', '0', '0', '0', '1', '1', '0',
'0', '0', '0', '1', '0', '0', '1', '1', '0', '0', '1', '0', '1', '1', '1', '1',
'0', '1', '1', '0', '1', '0', '1', '0', '1', '1', '0', '1', '0', '0', '1', '0',
'1', '0', '0', '0', '0', '1', '1', '0', '0', '0', '0', '0', '1', '0', '1', '0',
'0', '1', '1', '1', '0', '0', '0', '1', '0', '0', '0', '0', '0', '0', '0', '0',
'0', '0', '1', '1', '1', '0', '0', '0', '0', '0', '1', '0', '0', '0', '1', '0',
'1', '1', '1', '0', '1', '1', '1', '1', '1', '0', '1', '1', '1', '0', '1', '0',
'1', '1', '0', '1', '0', '1', '1', '1', '1', '1', '0', '0', '1', '1', '0', '0',
'1', '0', '1', '1', '1', '1', '1', '1', '1', '1', '1', '0', '0', '1', '1', '0',
'0', '1', '1', '1', '0', '1', '1', '1', '0', '1', '0', '0', '1', '1', '0', '0',
'0', '0', '1', '1', '1', '1', '0', '0', '1', '0', '0', '1', '1', '1', '0', '0',
'0', '0', '0', '0', '1', '1', '1', '0', '1', '0', '0', '0', '1', '0', '0', '1',

'1', '1', '1', '0', '0', '0', '0', '0', '1', '1', '1', '0', '0', '0', '0', '0',
'0', '1', '0', '0', '1', '0', '0', '1', '1', '1', '1', '1', '1', '1', '1', '0',
'0', '1', '0', '0', '1', '0', '1', '1', '1', '1', '1', '0', '1', '1', '0', '0',
'0', '0', '1', '0', '0', '0', '0', '0', '1', '0', '1', '0', '0', '0', '1', '1',
'0', '1', '0', '1', '1', '0', '1', '0', '1', '1', '0', '0', '0', '0', '1', '0',
'0', '1', '1', '1', '0', '0', '1', '0', '1', '0', '0', '0', '1', '0', '1', '1',
'0', '1', '0', '1', '0', '0', '1', '1', '0', '1', '1', '1', '1', '1', '0', '1',
'0', '0', '0', '1', '1', '1', '1', '1', '0', '0', '0', '1', '1', '0', '1', '1',
'0', '0', '1', '0', '1', '0', '1', '0', '0', '1', '0', '0', '1', '0', '0', '1',
'1', '0', '1', '0', '1', '1', '0', '0', '0', '0', '1', '1', '1', '0', '1', '0',
'0', '1', '1', '1', '1', '0', '0', '0', '1', '1', '1', '0', '1', '0', '1', '1',
'1', '1', '0', '0', '0', '0', '1', '1', '1', '1', '0', '0', '1', '1', '0', '0',
'0', '0', '1', '0', '0', '1', '0', '0', '1', '0', '0', '0', '1', '0', '0', '1',
'1', '0', '1', '0', '1', '1', '0', '0', '0', '0', '0', '0', '0', '0', '1', '1',
'1', '0', '1', '1', '0', '1', '1', '1', '0', '0', '1', '1', '1', '0', '1', '0',
'1', '1', '1', '1', '0', '0', '1', '0', '0', '0', '1', '0', '0', '1', '1', '1',
'0', '0', '0', '0', '0', '0', '1', '0', '0', '0', '1', '0', '1', '1', '0', '0',
'1', '1', '1', '1', '0', '1', '1', '0', '0', '1', '0', '1', '1', '0', '0', '0',
'1', '0', '0', '0', '1', '0', '0', '1', '0', '1', '0', '0', '1', '0', '0', '0',
'1', '1', '1', '0', '0', '1', '0', '0', '0', '0', '0', '0', '0', '1', '1', '1',
'0', '1', '0', '1', '1', '1', '0', '0', '0', '0', '0', '0', '1', '0', '1', '1',
'1', '1', '0', '1', '1', '0', '1', '0', '0', '1', '0', '1', '1', '1', '0', '1',
'1', '0', '0', '1', '0', '0', '1', '1', '1', '1', '1', '0', '0', '1', '0', '1',
'0', '1', '0', '1', '0', '0', '1', '1', '1', '0', '0', '1', '1', '1', '0', '0',
'0', '1', '0', '0', '0', '1', '0', '0', '0', '0', '1', '0', '0', '1', '1', '0',
'0', '1', '1', '0', '1', '0', '0', '1', '1', '0', '0', '0', '0', '0', '0', '1',
'1', '0', '1', '1', '0', '0', '0', '0', '1', '1', '0', '1', '0', '0', '1', '0',
'0', '0', '1', '0', '0', '1', '0', '1', '1', '1', '0', '1', '0', '0', '1', '0',
'1', '1', '0', '0', '0', '0', '1', '0', '0', '0', '1', '0', '1', '1', '0', '0',
'0', '0', '0', '1', '1', '0', '1', '0', '0', '1', '0', '0', '1', '0', '1', '1',
'1', '1', '0', '1', '1', '1', '1', '1', '1', '1', '1', '1', '1', '0', '0', '1',
'1', '1', '0', '0', '1', '1', '1', '0', '1', '0', '1', '0', '0', '1', '1', '1',
'1', '1', '0', '0', '0', '0', '0', '1', '0', '0', '0', '0', '0', '0', '1', '0',
'0', '0', '0', '0', '0', '1', '0', '0', '1', '0', '1', '0', '0', '0', '0', '0',
'0', '0', '0', '0', '0', '1', '1', '1', '1', '0', '1', '0', '0', '1', '0', '0',
'0', '0', '1', '0', '1', '0', '0', '0', '0', '1', '0', '1', '1', '1', '1', '0',
'1', '1', '0', '1', '0', '1', '1', '1', '1', '0', '0', '1', '1', '1', '1', '0',
'0', '0', '0', '1', '1', '1', '1', '0', '0', '1', '0', '1', '0', '0', '0', '0',
'1', '1', '1', '1', '0', '0', '0', '1', '0', '1', '1', '1', '0', '1', '0', '1',
'0', '0', '1', '1', '1', '1', '0', '0', '0', '0', '0', '1', '0', '1', '1', '1',
'1', '0', '0', '0', '0', '1', '0', '1', '0', '1', '1', '1', '1', '0', '0', '0',
'0', '0', '0', '0', '1', '0', '1', '0', '1', '1', '0', '1', '0', '1', '0', '0',
'1', '0', '0', '0', '0', '0', '0', '1', '0', '1', '0', '1', '0', '0', '1', '1',
'0', '1', '1', '1', '1', '1', '0', '1', '1', '1', '1', '1', '1', '0', '0', '1', '0',
'1', '0', '1', '1', '0', '1', '0', '0', '0', '1', '1', '1', '1', '1', '1', '0',
'0', '1', '1', '0', '1', '0', '1', '0', '0', '0', '0', '1', '0', '1', '1', '0',
'0', '1', '1', '0', '0', '0', '1', '1', '1', '0', '0', '1', '1', '1', '1', '0',
'0', '0', '0', '1', '1', '1', '1', '0', '1', '1', '0', '1', '1', '1', '0', '0',

```
'0', '0', '1', '1', '0', '0', '1', '1', '1', '0', '1', '0', '0', '1', '1', '0',
'0', '1', '1', '0', '1', '1', '1', '0', '1', '0', '0', '0', '0', '1', '0', '0',
'1', '0', '0', '0', '0', '1', '0', '1', '1', '0', '0', '1', '0', '0', '0', '1',
'0', '1', '1', '0', '0', '0', '0', '1', '1', '1', '0', '0', '0', '1', '0', '0',
'1', '0', '0', '0', '1', '0', '0', '0', '0', '0', '0', '1', '0', '1', '0', '0',
'0', '1', '0', '0', '1', '1', '0', '1', '0', '1', '1', '1', '0', '0', '1', '0',
'1', '1', '1', '0', '1', '0', '0', '1', '0', '0', '1', '1', '0', '1', '1', '1',
'0', '0', '1', '0', '1', '0', '0', '0', '0', '0', '1', '1', '1', '1', '0', '0',
'0', '1', '0', '0', '1', '1', '1', '0', '1', '0', '0', '1', '0']
```

[16]:
```
test_4c( key, alice_random_trits, bob_random_trits  )
key
```

```
La clave no es correcta
```

[16]:
```
['0',
 '0',
 '0',
 '0',
 '1',
 '0',
 '0',
 '0',
 '0',
 '1',
 '1',
 '1',
 '1',
 '0',
 '1',
 '1',
 '1',
 '1',
 '0',
 '0',
 '0',
 '1',
 '0',
 '1',
 '1',
 '0',
 '1',
 '1',
 '0',
 '1',
 '0',
 '1',
 '1',
```

```
'1',
'1',
'1',
'1',
'1',
'1',
'0',
'0',
'1',
'1',
'1',
'0',
'0',
'1',
'0',
'1',
'0',
'0',
'1',
'1',
'0',
'1',
'1',
'0',
'0',
'0',
'0',
'1',
'0',
'1',
'1',
'1',
'0',
'0',
'0',
'1',
'0',
'0',
'1',
'0',
'1',
'0',
'1',
'1',
'0',
'0',
'0',
```

```
'0',
'0',
'0',
'1',
'1',
'0',
'1',
'1',
'1',
'0',
'0',
'1',
'1',
'1',
'0',
'1',
'0',
'0',
'0',
'1',
'0',
'1',
'1',
'0',
'0',
'1',
'0',
'1',
'0',
'0',
'0',
'0',
'1',
'0',
'1',
'1',
'0',
'0',
'0',
'0',
'1',
'0',
'0',
'0',
'0',
'1',
'1',
```

```
'1',
'1',
'1',
'0',
'1',
'0',
'1',
'0',
'1',
'1',
'1',
'1',
'1',
'1',
'0',
'0',
'1',
'1',
'1',
'1',
'0',
'0',
'1',
'0',
'0',
'1',
'0',
'0',
'0',
'0',
'1',
'1',
'1',
'1',
'0',
'0',
'1',
'1',
'1',
'0',
'1',
'1',
'0',
'1',
'0',
'1',
'1',
'0',
'1',
```

```
'1',
'0',
'0',
'1',
'1',
'1',
'0',
'1',
'0',
'0',
'1',
'1',
'0',
'0',
'0',
'0',
'1',
'0',
'1',
'1',
'0',
'0',
'0',
'1',
'0',
'1',
'0',
'1',
'0',
'0',
'0',
'0',
'1',
'0',
'0',
'0',
'1',
'0',
'0',
'1',
'0',
'0',
'0',
'1',
'0',
'1',
'0',
```

```
'0',
'1',
'0',
'0',
'0',
'1',
'0',
'0',
'0',
'1',
'0',
'1',
'0',
'0',
'0',
'0',
'1',
'0',
'1',
'1',
'1',
'0',
'1',
'1',
'1',
'1',
'1',
'0',
'1',
'0',
'0',
'0',
'1',
'0',
'1',
'1',
'0',
'1',
'0',
'0',
'1',
'1',
'0',
'1',
'0',
'0',
'1',
```

```
'0',
'0',
'0',
'0',
'1',
'0',
'1',
'1',
'0',
'0',
'1',
'1',
'0',
'0',
'1',
'1',
'1',
'0',
'0',
'1',
'1',
'1',
'0',
'0',
'1',
'1',
'1',
'1',
'1',
'1',
'0',
'1',
'1',
'1',
'1',
'1',
'0',
'0',
'1',
'1',
'1',
'1',
'1',
'0',
'1',
'1',
'0',
'0',
```

```
'1',
'1',
'1',
'0',
'0',
'0',
'1',
'0',
'0',
'1',
'0',
'0',
'0',
'1',
'1',
'0',
'1',
'0',
'0',
'0',
'1',
'1',
'0',
'1',
'0',
'1',
'0',
'0',
'0',
'0',
'1',
'1',
'1',
'1',
'0',
'0',
'0',
'1',
'1',
'1',
'0',
'1',
'0',
'0',
'1',
'1',
'0',
```

```
'1',
'1',
'0',
'1',
'1',
'1',
'0',
'1',
'1',
'0',
'1',
'1',
'0',
'1',
'0',
'0',
'0',
'0',
'0',
'0',
'1',
'0',
'1',
'1',
'1',
'1',
'0',
'1',
'0',
'0',
'1',
'0',
'1',
'1',
'0',
'0',
'0',
'0',
'1',
'1',
'0',
'1',
'1',
'0',
'0',
'1',
'1',
```

```
'0',
'0',
'1',
'1',
'0',
'1',
'0',
'0',
'1',
'1',
'1',
'0',
'1',
'0',
'1',
'0',
'1',
'1',
'1',
'1',
'1',
'1',
'0',
'0',
'0',
'1',
'0',
'1',
'0',
'1',
'1',
'0',
'0',
'0',
'0',
'0',
'0',
'0',
'0',
'1',
'0',
'0',
'1',
'1',
'0',
'0',
'1',
'0',
```

```
'0',
'0',
'1',
'1',
'1',
'0',
'1',
'1',
'1',
'0',
'0',
'1',
'0',
'0',
'0',
'1',
'1',
'0',
'0',
'1',
'0',
'1',
'1',
'0',
'0',
'1',
'1',
'0',
'1',
'0',
'0',
'1',
'1',
'1',
'0',
'1',
'0',
'1',
'1',
'1',
'1',
'1',
'0',
'0',
'1',
'0',
'1',
```

```
'1',
'0',
'1',
'0',
'1',
'1',
'1',
'0',
'1',
'1',
'1',
'0',
'0',
'1',
'0',
'1',
'1',
'0',
'1',
'0',
'0',
'0',
'0',
'0',
'0',
'0',
'0',
'0',
'1',
'1',
'1',
'0',
'1',
'0',
'1',
'1',
'0',
'0',
'0',
'1',
'1',
'0',
'0',
'0',
'1',
'1',
'1',
```

```
'0',
'1',
'0',
'1',
'1',
'1',
'1',
'1',
'0',
'0',
'1',
'1',
'1',
'1',
'0',
'0',
'1',
'0',
'0',
'0',
'0',
'1',
'0',
'1',
'1',
'0',
'0',
'1',
'0',
'0',
'0',
'0',
'0',
'1',
'1',
'0',
'0',
'1',
'1',
'0',
'1',
'0',
'1',
'1',
'1',
'0',
'1',
```

```
'1',
'0',
'1',
'0',
'1',
'1',
'0',
'1',
'0',
'0',
'1',
'0',
'1',
'1',
'0',
'1',
'1',
'1',
'1',
'0',
'0',
'1',
'0',
'0',
'0',
'0',
'1',
'0',
'1',
'1',
'1',
'0',
'0',
'1',
'0',
'1',
'1',
'0',
'0',
'0',
'0',
'0',
'1',
'1',
'0',
'0',
'0',
```

```
'0',
'0',
'1',
'0',
'0',
'1',
'1',
'0',
'1',
'1',
'0',
'0',
'1',
'0',
'0',
'0',
'1',
'0',
'1',
'1',
'0',
'1',
'0',
'0',
'0',
'1',
'0',
'0',
'0',
'0',
'0',
'1',
'0',
'1',
'0',
'0',
'1',
'1',
'0',
'0',
'1',
'0',
'0',
'0',
'1',
'1',
'0',
```

```
'1',
'0',
'0',
'1',
'0',
'1',
'0',
'1',
'1',
'1',
'0',
'0',
'0',
'0',
'1',
'0',
'0',
'1',
'1',
'1',
'0',
'1',
'1',
'0',
'1',
'0',
'1',
'0',
'1',
'0',
'1',
'1',
'1',
'1',
'1',
'1',
'1',
'0',
'0',
'1',
'0',
'0',
'1',
'1',
'0',
'1',
'1',
'1',
```

```
'1',
'0',
'0',
'0',
'1',
'1',
'0',
'1',
'0',
'0',
'1',
'0',
'0',
'1',
'1',
'1',
'1',
'0',
'1',
'0',
'1',
'0',
'0',
'1',
'0',
'1',
'0',
'1',
'0',
'0',
'1',
'0',
'1',
'0',
'1',
'1',
'1',
'1',
'0',
'1',
'0',
'0',
'0',
'1',
'1',
```

```
'1',
'0',
'0',
'1',
'0',
'1',
'0',
'1',
'1',
'0',
'1',
'1',
'0',
'0',
'0',
'1',
'0',
'1',
'1',
'1',
'1',
'0',
'0',
'0',
'0',
'0',
'0',
'0',
'0',
'0',
'0',
'1',
'1',
'1',
'0',
'1',
'0',
'0',
'0',
'1',
'1',
'0',
'0',
'1',
'1',
'1',
'0',
```

```
'0',
'1',
'0',
'0',
'1',
'1',
'0',
'0',
'1',
'0',
'0',
'0',
'0',
'1',
'0',
'1',
'1',
'1',
'0',
'1',
'1',
'0',
'1',
'0',
'0',
'1',
'1',
'1',
'0',
'0',
'0',
'0',
'0',
'0',
'1',
'1',
'0',
'1',
'0',
'0',
'1',
'1',
'1',
'0',
'0',
'0',
'1',
```

```
'0',
'0',
'1',
'1',
'1',
'0',
'1',
'0',
'1',
'1',
'1',
'1',
'1',
'0',
'1',
'1',
'0',
'1',
'0',
'1',
'0',
'1',
'0',
'1',
'1',
'1',
'1',
'0',
'0',
'0',
'0',
'0',
'1',
'1',
'0',
'0',
'1',
'0',
'1',
'1',
'1',
'1',
'1',
'1',
'0',
'1',
'0',
'0',
```

```
'1',
'0',
'0',
'1',
'0',
'1',
'1',
'1',
'1',
'1',
'1',
'0',
'0',
'1',
'0',
'1',
'0',
'1',
'0',
'0',
'0',
'1',
'0',
'0',
'0',
'1',
'0',
'0',
'0',
'1',
'0',
'1',
'1',
'0',
'0',
'1',
'0',
'1',
'1',
'0',
'0',
'1',
'0',
'1',
'0',
'0',
'0',
```

```
 '0',
 '0',
 '0',
 '0',
 '1',
 '0',
 '0',
 '1',
 '0',
 '1',
 '0',
 '1',
 '1',
 '0',
 '1',
 '1',
 '0',
 '0',
 '0',
 '1',
 '1',
 '1',
 '0',
 '1',
 '1',
 '0',
 '0',
 …]
```

1.1 Este desafío fue diseñado por:

Alejandro Rojas Estudiante de doctorado en física Universidad de Concepción alejarojas@udec.cl

Luciano Pereira Postdoctoral researcher, ICFO, Spain luciano.pereira@icfo.es

[ ]: