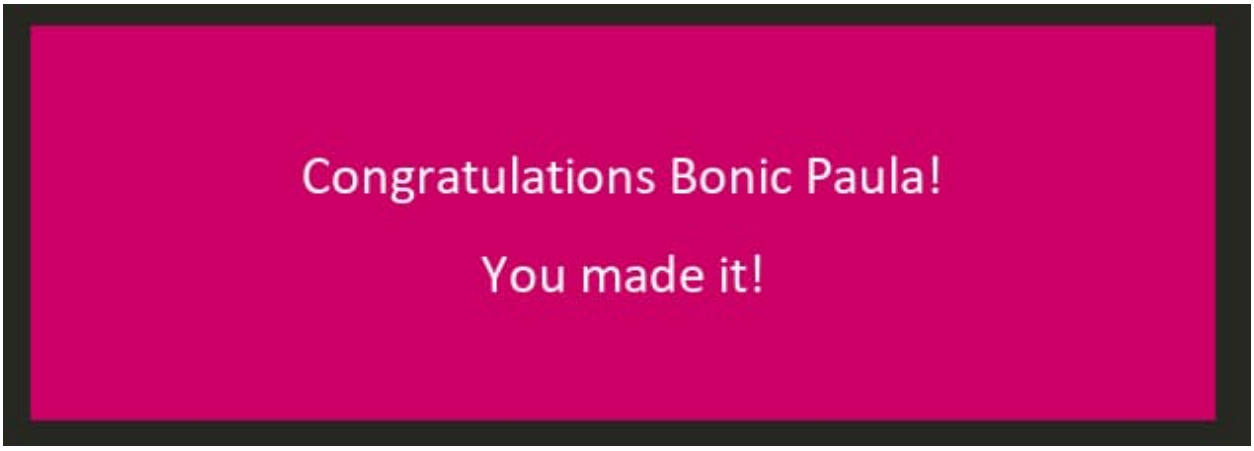
# Sigurnost računala i podataka - Lab 3.

Trebali smo riješiti kriptografski izazov. U zadanoj mapi su se nalazilo 10ak datoteka čiji su nazivi bili hashirana imena ljudi koji prisustvuju labovima. Prvo smo trebali otkriti koja je naša datoteka. Način na koji smo to otkrili je bilo hashiranje našeg imena. Zatim smo uspoređivali s imenima datoteka. Kad smo našli našu datoteku u njoj se nalazila vrijednost koja je na kraju predstavljala sliku. U metapodacima slike su zapisane informacije o slici. Za png format znamo da je prvih 8 bajtova karakterističnо tako da se koristimo s tim da dođemo do rješenja. Ključ smo tražili brute-force napadom odosno iterirali smo sve moguće ključeve. Rezultat može doći kad se testira pola mogućih ključeva. Ono bitno je da dobro dekodiramo tu sliku odnosno da ima smisla. Rezultat koji smo na kraju dobili je bio ovaj:




Kod:

```
import base64
from os import path
from cryptography.hazmat.primitives import hashes
from cryptography.fernet import Fernet

def hash(input):
    if not isinstance(input, bytes):
        input = input.encode()
```

```
    digest = hashes.Hash(hashes.SHA256())
    digest.update(input)
    hash = digest.finalize()

    return hash.hex()

def test_png(header):
    if header.startswith(b"\211PNG\r\n\032\n"):
        return True
    return False


def brute_force(ciphertext):
    ctr = 0
    while True:
        key_bytes = ctr.to_bytes(32, "big")
        key = base64.urlsafe_b64encode(key_bytes)


        # Now initialize the Fernet system with the given key
        # and try to decrypt your challenge.
        # Think, how do you know that the key tested is the correct key
        # (i.e., how do you break out of this infinite loop)?


        try:
            plaintext = Fernet(key).decrypt(ciphertext)
            header = plaintext[:32]

            if test_png(header):
                print(f"BINGO: {key}")
                with open("BINGO.png", "wb") as file:
                    file.write(plaintext)
                break
        except Exception:
            pass


        ctr += 1
        if not ctr % 1000:
            print(f"[*] Keys tested: {ctr:,}", end="\r")



if __name__ == "__main__":
    filename = hash("bonic_paula") + ".encrypted"

    # Create a file with the filename if it does not alread exist
    if not path.exists(filename):
        with open(filename, "wb") as file:
            file.write(b"")
```

```
#Open your challenge file and read in your challenge
with open(filename, "rb") as file:
    ciphertext = file.read()

# print(ciphertext)
# Start the attack
brute_force(ciphertext)
```