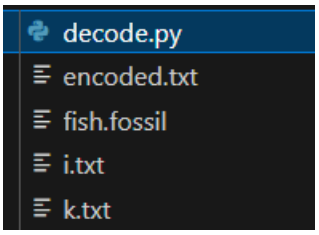
This blows is an easy debugging/scripting challenge. There are two approaches to solve this problem. Understand how it works and break it into fundamentals (Not using scripting) or debug the script. There are 4 files attached let's go through the two methods

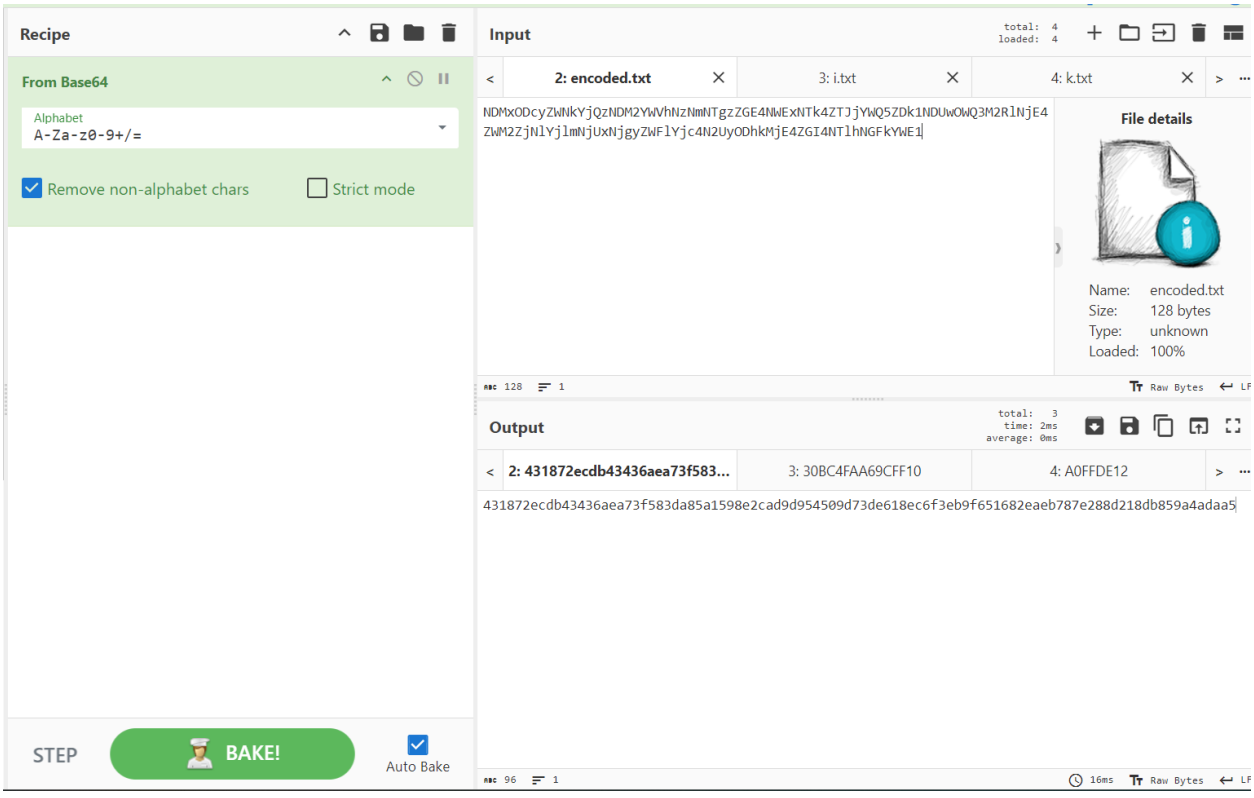


1 (No Scripting):

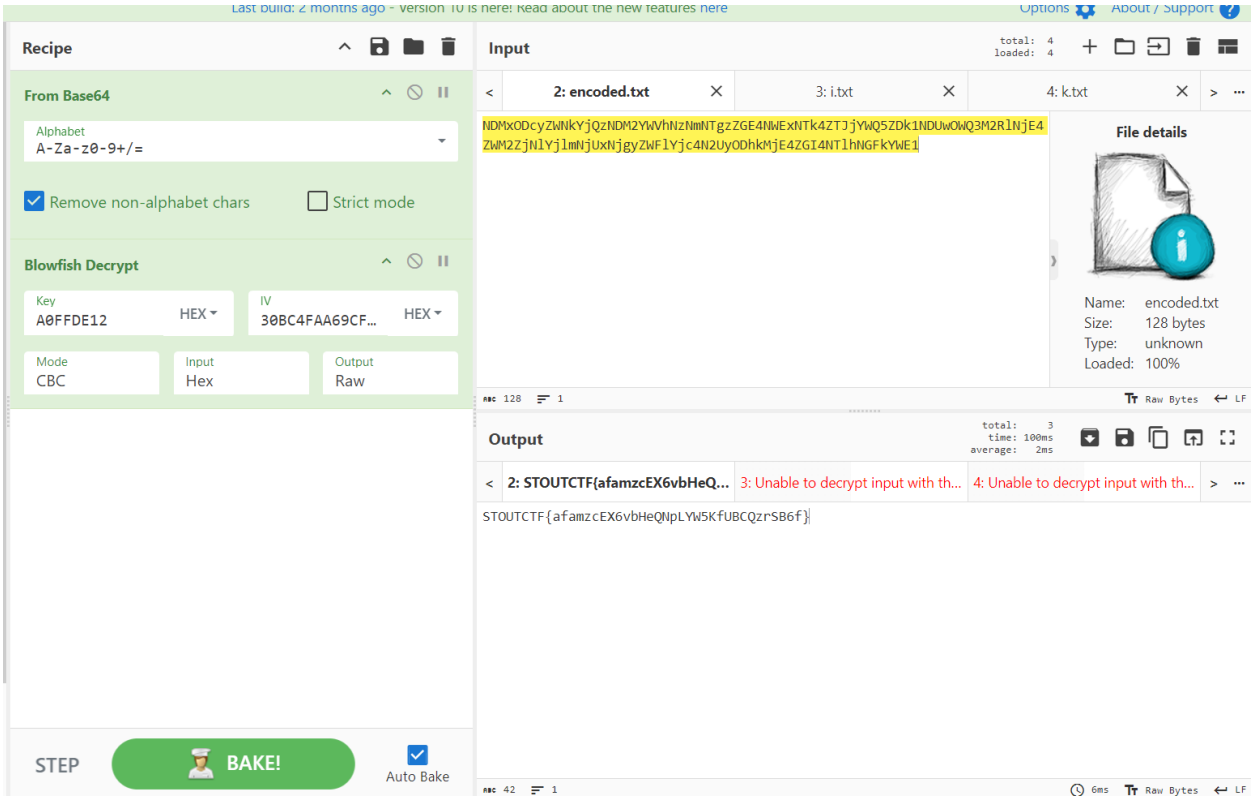
From the imports you can see there is going to be base 64 and blowfish encoding

```
#!/usr/bin/env python
import base64
from Crypto.Cipher import Blowfish
from Crypto.Random import get_random_bytes
```

Blowfish requires a Key and and IV. From the files before you have the encoded file. Cyber Chef can tell you its B64 encoded and then encrypted using Blowfish. This is what has to be decrypted. This can also be seen in the code. You also have files i.txt and k.txt. This is most likely the IV and the Key. Bring these into cyber chef.



Now decrypt using blowfish what we suspect to be the Key and the IV.



2 (Scripting):

Scripting is just as easy. Just understand what the different lines are doing. One thing that stands out is the Obfuscation, you might want to change the variable names to make more sense of the challenge. One error you can see if the b64 function is encoding and not decoding, change that. Another thing you can see if that there was a misspelt variable so it wasn't being called. You can also see that some of the code was commented out. You can also remove the decoding of fish.fossil though this is not required. Your code should run now and successfully decode. It should look something like this:

```
#!/usr/bin/env python
import base64
from Crypto.Cipher import Blowfish
from Crypto.Random import get_random_bytes

key = 0
iv = 0
encoded = 0

# To Do: fix the broken decode function
```

```
def decode(string):
    encodedString = string
    k = key
    i = iv

    ciphertext = Blowfish.new(k, Blowfish.MODE_CBC, i)
    h = cipher.decrypt(ciphertext)

    print(h)

# Parse files, possible issue
def parse():
    with open('k.txt', 'r') as f:
        k = f.read()

    key = b64(k) # a23 is a global variable which is decoded from b64

    with open('i.txt', 'r') as g:
        i = g.read()

    iv = b64(i)

    with open('encoded.txt', 'r') as h:
        z = h.read()

    encoded = b64(z)

# Decodes b64 to raw
def b64(string):
    d1 = base64.b64decode(string)
    return d1

def main():
    #parse
    parse()
    #decodes parsed data
    decode()
    # fishfossil() Not even a fucntion

#Runs the main function
if __name__ == "__main__":
    main()
```

Note:

You maybe asking what the fish.fossil file is. This is a useless file to through off chat GPT or any other generative AI. It sees the .fossil file and thinks that's where the solution is.