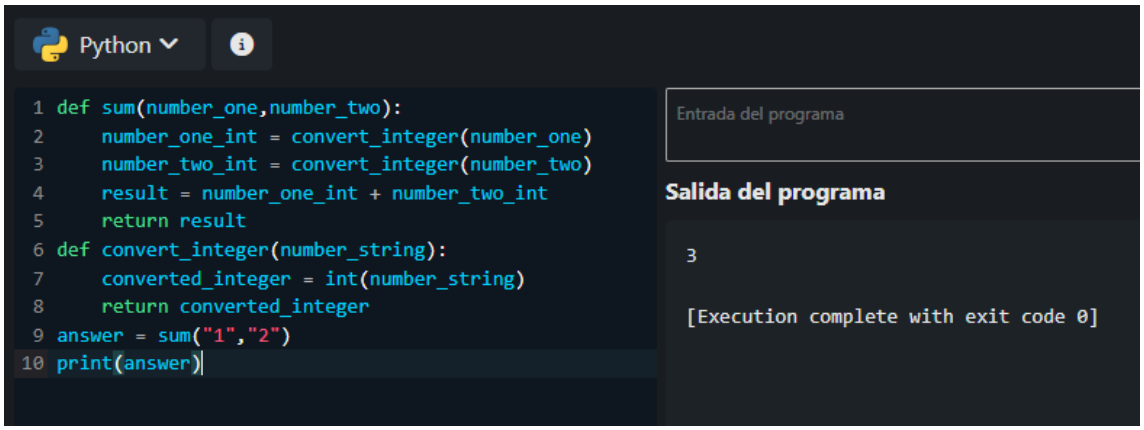
TRANSCRIPCIÓN SCRIPTS:

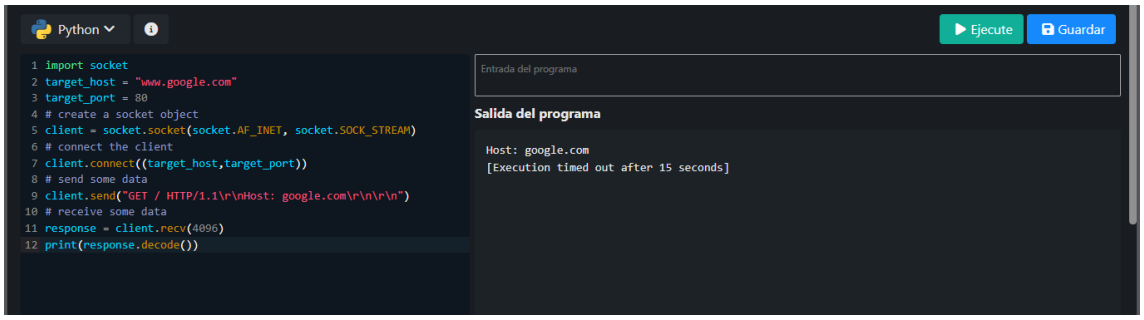
1. SCRIPT:

```python
def sum(number_one,number_two):
    number_one_int = convert_integer(number_one)
    number_two_int = convert_integer(number_two)
    result = number_one_int + number_two_int
    return result
def convert_integer(number_string):
    converted_integer = int(number_string)
    return converted_integer
answer = sum("1","2")
print(answer)
```



2. SCRIPT:

```python
import socket
target_host = "www.google.com"
target_port = 80
# create a socket object
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
# connect the client
client.connect((target_host,target_port))
# send some data
client.send("GET / HTTP/1.1\r\nHost: google.com\r\n\r\n")
# receive some data
response = client.recv(4096)
print(response.decode())
```



3. SCRIPT:

```python
import socket

target_host = "www.google.com"
target_port = 80

# create a socket object
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# send some data
client.connect((target_host, target_port))
request = b"AAABBBCCC"
client.send(request)

# receive some data
data = client.recv(4096)
print(data)

```

Python

```python
import socket

target_host = "www.google.com"
target_port = 80

# create a socket object
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# send some data
client.connect((target_host, target_port))
request = b"AAABBBCCC"
client.send(request)

# receive some data
data = client.recv(4096)
print(data)

```

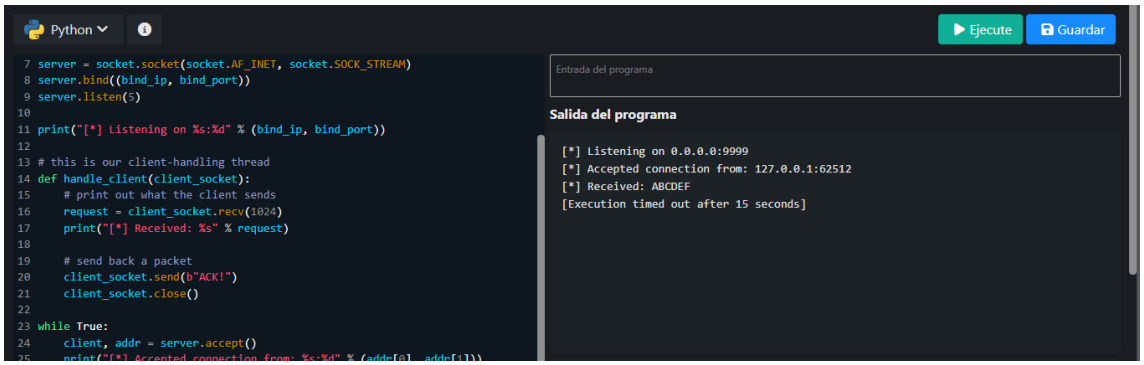
Entrada del programa

Salida del programa

```
Host: google.com
[Execution timed out after 15 seconds]
```

4. SCRIPT:

```python
import socket
import threading
bind_ip = "0.0.0.0"
bind_port = 9999
server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind((bind_ip,bind_port))
server.listen(5)
print("[*] Listening on %s:%d" % (bind_ip,bind_port))
# this is our client-handling thread
def handle_client(client_socket):
    # print out what the client sends
    request = client_socket.recv(1024)
    print("[*] Received: %s" % request)
    # send back a packet
    client_socket.send("ACK!")
    client_socket.close()
while True:
    client,addr = server.accept()
    print("[*] Accepted connection from: %s:%d" % (addr[0],addr[1]))
    # spin up our client thread to handle incoming data
    client_handler = threading.Thread(target=handle_client,args=(client,))
    client_handler.start()
```



5. SCRIPT:

```
import sys
import socket
import getopt
import threading
import subprocess

# define some global variables
listen = False
command = False
upload = False
execute = ""
target = ""
upload_destination = ""
port = 0

def usage():
    print("BHP Net Tool")
    print()
    print("Usage: bhpnet.py -t target_host -p port")
    print("-l --listen - listen on [host]:[port] for incoming connections")
    print("-e --execute=file_to_run - execute the given file upon receiving a connection")
    print("-c --command - initialize a command shell")
    print("-u --upload=destination - upon receiving connection upload a file and write to [destination]")
    print()
```

```
    print("Examples: ")
    print("bhpnet.py -t 192.168.0.1 -p 5555 -l -c")
    print("bhpnet.py -t 192.168.0.1 -p 5555 -l -u=c:\\target.exe")
    print("bhpnet.py -t 192.168.0.1 -p 5555 -l -e=\"cat /etc/passwd\"")
    print("echo 'ABCDEFGHI' | ./bhpnet.py -t 192.168.11.12 -p 135")
    sys.exit(0)

def main():
    global listen
    global port
    global execute
    global command
    global upload_destination
    global target

    if not len(sys.argv[1:]):
        usage()

    # read the commandline options
    try:
        opts, args = getopt.getopt(sys.argv[1:], "hle:t:p:cu:", ["help", "listen", "execute", "target", "port
    except getopt.GetoptError as err:
        print(str(err))
        usage()
```

```python
    for o, a in opts:
        if o in ("-h", "--help"):
            usage()
        elif o in ("-l", "--listen"):
            listen = True
        elif o in ("-e", "--execute"):
            execute = a
        elif o in ("-c", "--command"):
            command = True
        elif o in ("-u", "--upload"):
            upload_destination = a
        elif o in ("-t", "--target"):
            target = a
        elif o in ("-p", "--port"):
            port = int(a)
        else:
            assert False, "Unhandled Option"

    # are we going to listen or just send data from stdin?
    if not listen and len(target) and port > 0:
        # read in the buffer from the commandline
        # this will block, so send CTRL-D if not sending input to stdin
        buffer = sys.stdin.read()
        # send data off
```

```python
        client_sender(buffer)

    # we are going to listen and potentially upload things, execute commands, and drop a shell back depending
    if listen:
        server_loop()

main()
```

Python

Ejecute Guardar

```python
        elif o in ("-c", "--command"):
            command = True
        elif o in ("-u", "--upload"):
            upload_destination = a
        elif o in ("-t", "--target"):
            target = a
        elif o in ("-p", "--port"):
            port = int(a)
        else:
            assert False, "Unhandled Option"

    # are we going to listen or just send data from stdin?
    if not listen and len(target) and port > 0:
        # read in the buffer from the commandline
        # this will block, so send CTRL-D if not sending input
        buffer = sys.stdin.read()
        # send data off
        client_sender(buffer)
```

Entrada del programa

Salida del programa

```
-l --listen - listen on [host]:[port] for incoming connections
-e --execute=file_to_run - execute the given file upon receiving a connection
-c --command - initialize a command shell
-u --upload=destination - upon receiving connection upload a file and write to [destination]


Examples:
bhpnet.py -t 192.168.0.1 -p 5555 -l -c
bhpnet.py -t 192.168.0.1 -p 5555 -l -u=c:\target.exe
bhpnet.py -t 192.168.0.1 -p 5555 -l -e="cat /etc/passwd"
echo 'ABCDEFGHI' | ./bhpnet.py -t 192.168.11.12 -p 135

[Execution complete with exit code 0]
```