Assignment 5

COMPUTER NETWORK LAB
NAME = ANIK BARURY
ROLL = CSE22017
REGISTRATION NO = 871

1) To write a TCP socket program (in C/C++/Java/Python) to establish connection between client and server. The client program will send an input string to the server and the server program will check whether the input string is palindrome or not. Client will display the value send by server. The communication between client and server will continue until client send 'Quit' message to the server.

Server.py ==>

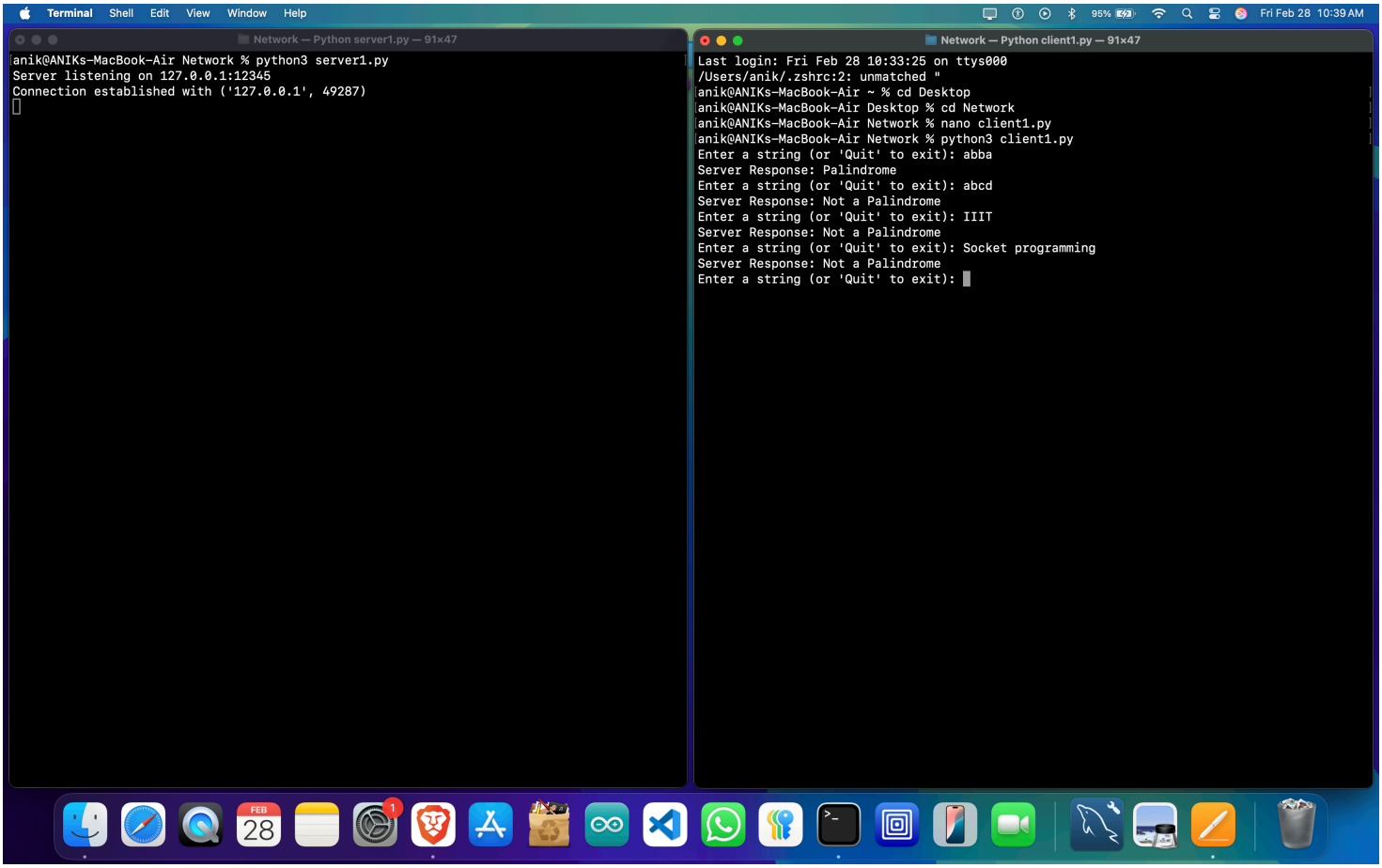
```
import socket

def is_palindrome(s):
    return s == s[::-1]

def start_server(host='127.0.0.1', port=12345):
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server_socket.bind((host, port))
    server_socket.listen(1)
    print(f"Server listening on {host}:{port}")

    conn, addr = server_socket.accept()
    print(f"Connection established with {addr}")

    while True:
        data = conn.recv(1024).decode()
        if not data or data.lower() == 'quit':
            print("Client has disconnected.")
            break

        response = "Palindrome" if is_palindrome(data) else "Not a Palindrome"
        conn.send(response.encode())

    conn.close()
    server_socket.close()

if __name__ == "__main__":
    start_server()
```

Client.py ==>

```
import socket

def start_client(host='127.0.0.1', port=12345):
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
```

```
    client_socket.connect((host, port))

    while True:
        n = input("Enter a string (or 'Quit' to exit): ")
        client_socket.sendall(n.encode())

        if n.lower() == 'quit':
            break

        response = client_socket.recv(1024).decode()
        print(f"Server Response: {response}")

    client_socket.close()

if __name__ == "__main__":
    start_client()
```

Output ==>



2) To write a TCP socket program (in C/C++/Java/Python) to establish connection between client and server. The client program will send an input string to the server and the server program will send back the reverse of the input string to the client. Client will display the value send by server. The communication between client and server will continue until client send 'Quit' message to the server.

Server.py ==>

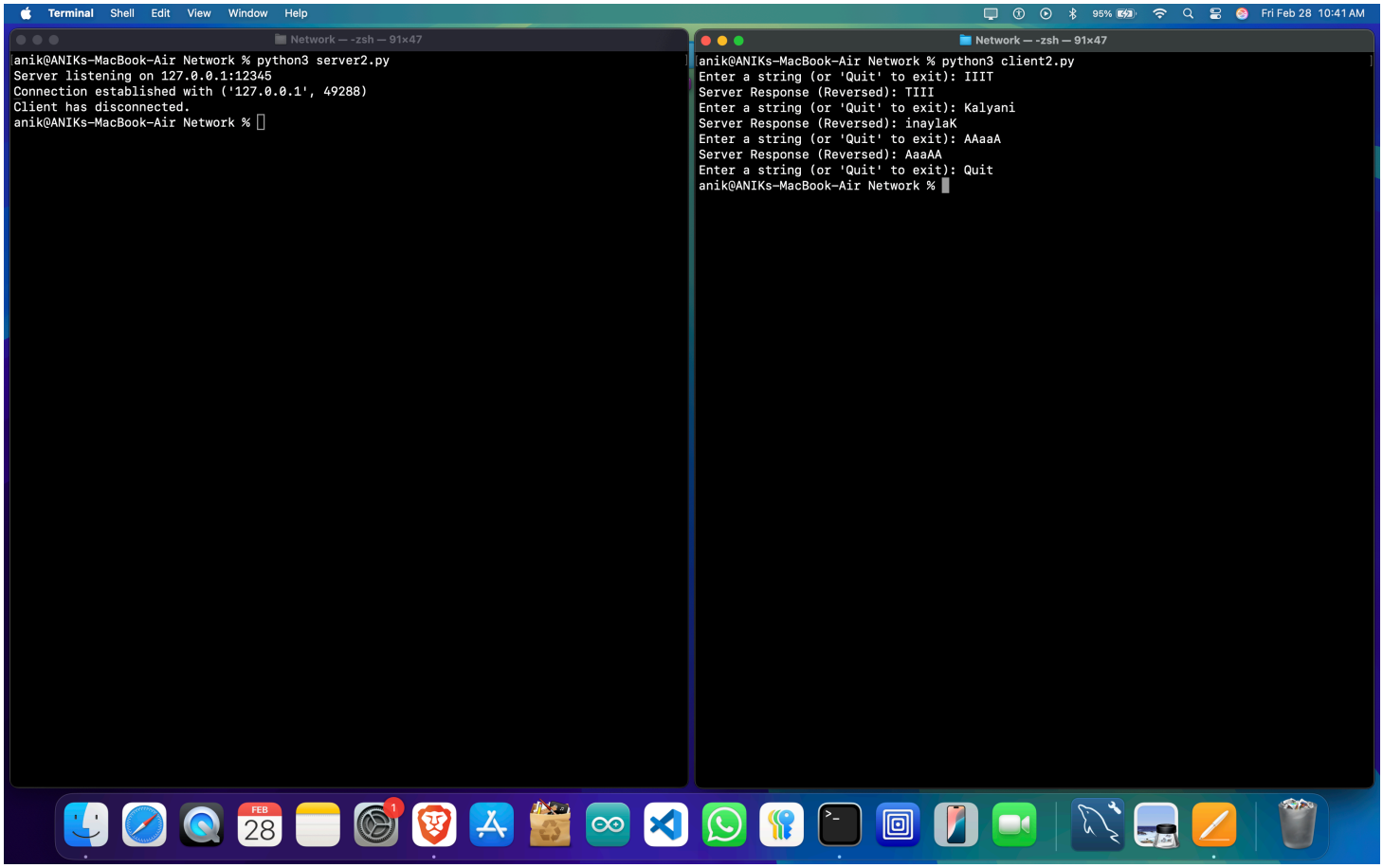
```
import socket

def reverse_string(s):
    return s[::-1]

def start_server(host='127.0.0.1', port=12345):
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server_socket.bind((host, port))
    server_socket.listen(1)
    print(f"Server listening on {host}:{port}")

    conn, addr = server_socket.accept()
    print(f"Connection established with {addr}")

    while True:
        data = conn.recv(1024).decode()
        if not data or data.lower() == 'quit':
            print("Client has disconnected.")
            break

        response = reverse_string(data)
        conn.send(response.encode())

    conn.close()
    server_socket.close()

if __name__ == "__main__":
    start_server()
```

Client.py ==>

```
import socket

def start_client(host='127.0.0.1', port=12345):
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client_socket.connect((host, port))
```

```
    while True:
        user_input = input("Enter a string (or 'Quit' to exit): ")
        client_socket.sendall(user_input.encode())

        if user_input.lower() == 'quit':
            break

        response = client_socket.recv(1024).decode()
        print(f"Server Response (Reversed): {response}")

    client_socket.close()

if __name__ == "__main__":
    start_client()
```

Output ==>



3) To write a TCP socket program (in C/C++/Java/Python) to establish connection between client and server. The client program will send send a URL to the server and a depth up to which the web-crawler visits all the pages from the initial page. Server runs a web crawler function up to the given depth to check all the URLs available and send the list of those URLs to Client. Client will display the list send by server. The communication between client and server will continue until client send 'Quit' message to the server.

Server.py ==>

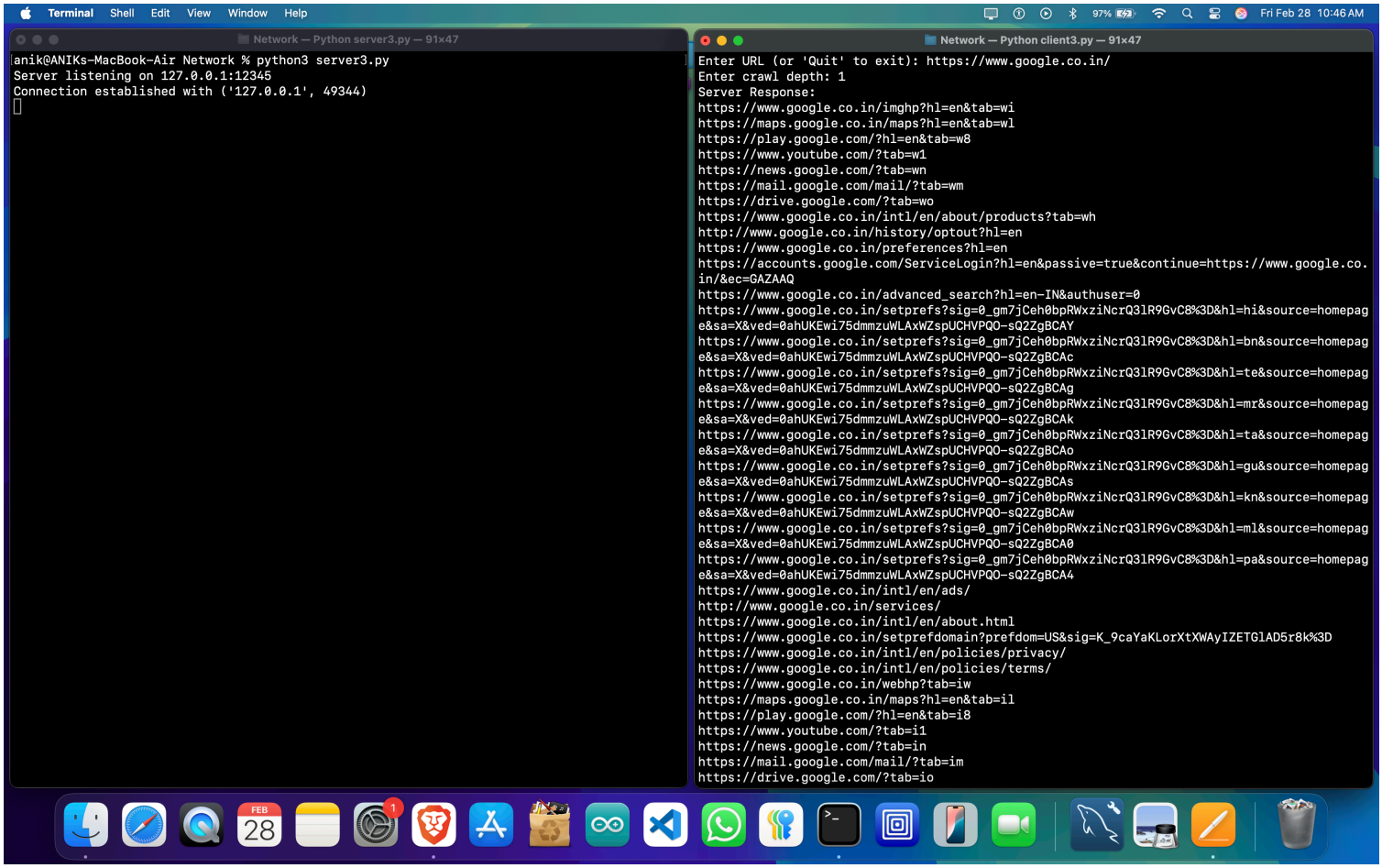
```
import socket
import requests
from bs4 import BeautifulSoup
from urllib.parse import urljoin

def crawl(url, depth):
    if depth <= 0:
        return []

    visited = set()
    urls_to_visit = [(url, 0)]
    found_urls = []

    while urls_to_visit:
        current_url, current_depth = urls_to_visit.pop(0)
        if current_url in visited or current_depth > depth:
            continue

        visited.add(current_url)
        try:
            response = requests.get(current_url, timeout=5)
            soup = BeautifulSoup(response.text, 'html.parser')

            for link in soup.find_all('a', href=True):
                full_url = urljoin(current_url, link['href'])
                if full_url not in visited:
                    urls_to_visit.append((full_url, current_depth + 1))
                    found_urls.append(full_url)
        except requests.exceptions.RequestException:
            continue

    return found_urls

def start_server(host='127.0.0.1', port=12345):
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server_socket.bind((host, port))
    server_socket.listen(1)
    print(f"Server listening on {host}:{port}")

    conn, addr = server_socket.accept()
    print(f"Connection established with {addr}")

    while True:
        data = conn.recv(1024).decode()
        if not data or data.lower() == 'quit':
            print("Client has disconnected.")
```

```
            break

        try:
            url, depth = data.split()
            depth = int(depth)
            urls = crawl(url, depth)
            response = '\n'.join(urls) if urls else "No URLs found."
        except Exception as e:
            response = f"Error: {str(e)}"

        conn.send(response.encode())

    conn.close()
    server_socket.close()

if __name__ == "__main__":
    start_server()
```

Client.py ==>

```
import socket

def start_client(host='127.0.0.1', port=12345, output_file="crawled_urls.txt"):
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client_socket.connect((host, port))

    while True:
        url = input("Enter URL (or 'Quit' to exit): ")
        if url.lower() == 'quit':
            client_socket.sendall(url.encode())
            break

        depth = input("Enter crawl depth: ")
        message = f"{url} {depth}"
        client_socket.sendall(message.encode())

        response = client_socket.recv(4096).decode()
        print(f"Server Response:\n{response}")

        with open(output_file, "w") as file:
            file.write(response)

        print(f"Results saved to {output_file}")

    client_socket.close()

if __name__ == "__main__":
    start_client()
```

Output ==>


Terminal Shell Edit View Window Help
Fri Feb 28 10:46 AM
Network — Python server3.py — 91×47
anik@ANIKs-MacBook-Air Network % python3 server3.py
Server listening on 127.0.0.1:12345
Connection established with ('127.0.0.1', 49344)
Network — Python client3.py — 91×47
Enter URL (or 'Quit' to exit): https://www.google.co.in/
Enter crawl depth: 1
Server Response:
https://www.google.co.in/imghp?hl=en&tab=wi
https://maps.google.co.in/maps?hl=en&tab=wl
https://play.google.com/?hl=en&tab=w8
https://www.youtube.com/?tab=w1
https://news.google.com/?tab=wn
https://mail.google.com/mail/?tab=wm
https://drive.google.com/?tab=wo
https://www.google.co.in/intl/en/about/products?tab=wh
http://www.google.co.in/history/optout?hl=en
https://www.google.co.in/preferences?hl=en
https://accounts.google.com/ServiceLogin?hl=en&passive=true&continue=https://www.google.co.in/&ec=GAZAAQ
https://www.google.co.in/advanced_search?hl=en-IN&authuser=0
https://www.google.co.in/setprefs?sig=0_gm7jCeh0bpRWxziNcrQ3lR9GvC8%3D&hl=hi&source=homepage&sa=X&ved=0ahUKEwi75dmmzuWLAxWZspUCHVPQO-sQ2ZgBCAY
https://www.google.co.in/setprefs?sig=0_gm7jCeh0bpRWxziNcrQ3lR9GvC8%3D&hl=bn&source=homepage&sa=X&ved=0ahUKEwi75dmmzuWLAxWZspUCHVPQO-sQ2ZgBCAc
https://www.google.co.in/setprefs?sig=0_gm7jCeh0bpRWxziNcrQ3lR9GvC8%3D&hl=te&source=homepage&sa=X&ved=0ahUKEwi75dmmzuWLAxWZspUCHVPQO-sQ2ZgBCAg
https://www.google.co.in/setprefs?sig=0_gm7jCeh0bpRWxziNcrQ3lR9GvC8%3D&hl=mr&source=homepage&sa=X&ved=0ahUKEwi75dmmzuWLAxWZspUCHVPQO-sQ2ZgBCAk
https://www.google.co.in/setprefs?sig=0_gm7jCeh0bpRWxziNcrQ3lR9GvC8%3D&hl=ta&source=homepage&sa=X&ved=0ahUKEwi75dmmzuWLAxWZspUCHVPQO-sQ2ZgBCAo
https://www.google.co.in/setprefs?sig=0_gm7jCeh0bpRWxziNcrQ3lR9GvC8%3D&hl=gu&source=homepage&sa=X&ved=0ahUKEwi75dmmzuWLAxWZspUCHVPQO-sQ2ZgBCAs
https://www.google.co.in/setprefs?sig=0_gm7jCeh0bpRWxziNcrQ3lR9GvC8%3D&hl=kn&source=homepage&sa=X&ved=0ahUKEwi75dmmzuWLAxWZspUCHVPQO-sQ2ZgBCAw
https://www.google.co.in/setprefs?sig=0_gm7jCeh0bpRWxziNcrQ3lR9GvC8%3D&hl=ml&source=homepage&sa=X&ved=0ahUKEwi75dmmzuWLAxWZspUCHVPQO-sQ2ZgBCA0
https://www.google.co.in/setprefs?sig=0_gm7jCeh0bpRWxziNcrQ3lR9GvC8%3D&hl=pa&source=homepage&sa=X&ved=0ahUKEwi75dmmzuWLAxWZspUCHVPQO-sQ2ZgBCA4
https://www.google.co.in/intl/en/ads/
http://www.google.co.in/services/
https://www.google.co.in/intl/en/about.html
https://www.google.co.in/setprefdomain?prefdom=US&sig=K_9caYaKLorXtXWAyIZETGlAD5r8k%3D
https://www.google.co.in/intl/en/policies/privacy/
https://www.google.co.in/intl/en/policies/terms/
https://www.google.co.in/webhp?tab=iw
https://maps.google.co.in/maps?hl=en&tab=il
https://play.google.com/?hl=en&tab=i8
https://www.youtube.com/?tab=i1
https://news.google.com/?tab=in
https://mail.google.com/mail/?tab=im
https://drive.google.com/?tab=io
FEB
28