```
#!/usr/bin/python
import socket
import sys
from urllib.parse import urlparse, parse_qs
# The server's hostname or IP address
host = "rtvm.cs.camosun.bc.ca"

# The port used by the server
port_1 = 80
port_2 = 10010

# The address of the webpages
url = sys.argv[1]
resource = urlparse(url).path

# create a socket called "s" in this format:
# s = socket.socket(addr_family, type)
# socket.AF_INET: IPv4
# socket.AF_INET6: IPv6
s1 = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s1.connect((host, port_1))

# must be exactly the same, including spaces
# send the following to the server. Once the server sees the empty
line, it sends back information
s1.send(("GET " + resource + " HTTP/1.1\r\n").encode("utf-8"))
s1.send(("Host: " + host + "\r\n").encode("utf-8"))
s1.send(("\r\n").encode("utf-8"))

# s2 is a socket used to send data to the server
s2 = socket.socket()
s2.connect((host, port_2))
ready_status = s2.recv(1024).decode("utf-8")

state = 1
while state != 4:
    # Read in headers from web server, looking for "<HTML>". when it
is read, move to state = 2
    # Store the readings in s1_date, split it into chunk_1, chunk_2
    if state == 1:
        s1_data = s1.recv(1024).decode("utf-8")
        if "<HTML>" in s1_data:
            # chunk_1: header, chunk_2: non-header
            chunk_1, chunk_2 = s1_data.split("<HTML>")

            # add "<HTML>" tag back to chunk_2 so that later the data
that client
            # sends to HTML2TEXT server is wihtin the HTML openning
and closing tags,
            # but don't send yet, move to state 2
```

```
            chunk_2 = "<HTML>" + chunk_2
            state = 2


    # in state 2, keep reading in block from web server, and sending
to html2text
    # server, when </HTML> is read, move to state = 3
    if state == 2:
        s2.send(chunk_2.encode("utf-8"))
        if "</HTML>" in chunk_2:
            state = 3

        # if not, keep reading from the server
        else:
            chunk_2 = s1.recv(1024).decode("utf-8")


    # if state == 3 - receive from html2text server, printing
    # received block as you go. when "ICS 226 HTML CONVERT COMPLETE"
    # is received, move to state 4
    if state == 3:
        chunk_returned = s2.recv(1024).decode("utf-8")
        if "ICS 226 HTML CONVERT COMPLETE" in chunk_returned:
            chunk_5, chunk_6 = chunk_returned.split("ICS 226 HTML
CONVERT COMPLETE")
            print(chunk_5, end = "")
            state = 4
        else:
            print(chunk_returned, end = "")

    if state == 4:
        s1.close()
        s2.close()
```