```
    1   import socket
    2   import datetime
    3   import sys
    4   import os
    5   from os import path
    6
    7   def less_than_three():
    8       if len(sys.argv) < 4 or len(sys.argv) > 4:
    9           sys.exit("ERROR: LESS OR MORE THAN THREE ↵
        ↳       PARAMETERS")
   10       else:
   11           pass
   12
   13   def get_ip():
   14       address = str(sys.argv[1])
   15       try:
   16           ip_address = socket.gethostbyname(address)
   17       except socket.gaierror:
   18           sys.exit("ERROR: HOST NAME DOES NOT EXIST OR IP ↵
        ↳       IS NOT WELL-FORMATTED")
   19       print("IP IS VALID") #just to check
   20       return address
   21
   22   def get_port():
   23       port = int(sys.argv[2])
   24       if port < 1024 or port > 64000:
   25           sys.exit("ERROR: PORT NUMBER MUST BE BETWEEN ↵
        ↳       1024 AND 64000 (INCLUSIVE)")
   26       else:
   27           print("PORT IS VALID") #just to check
   28       return port
   29
   30   def name_of_file():
   31       filename = str(sys.argv[3])
   32       if path.exists(filename):
   33           sys.exit("ERROR: FILE ALREADY EXISTS LOCALLY")
   34       else:
   35           print("FILE DOES NOT EXIST") #just to check
   36           return filename
```

```
37
38  def create_socket():
39      try:
40          s = socket.socket(socket.AF_INET, socket.SOCK_STREAM) #AF_INT is adress for IPV4, #SOCK_STREAM is socket type for TCP
41      except socket.error:
42          sys.exit("ERROR: FAILED TO CREATE SOCKET")
43      print("SOCKET CREATION SUCCESS") #just to check
44      return s
45
46  def connect(s, HOST, PORT):
47      try:
48          s.connect((HOST, PORT))
49      except socket.error:
50          s.close()
51          sys.exit("ERROR: CONNECT FAILURE")
52      print("CONNECTION SUCCESS") #just to check
53
54  def file_request(filename):
55      file = bytearray()
56      MagicNo = (0x497E).to_bytes(2, byteorder='big') #equivalent to 0x497E
57      Type = (1).to_bytes(1, byteorder='big')
58      FilenameLen = (len(filename)).to_bytes(2, byteorder='big')
59      Encoded_filename = filename.encode('utf-8') #returns utf-8 encoded version of the string
60
61      return bytearray(MagicNo + Type + FilenameLen + Encoded_filename)
62
63  def recieve_data(s):
64      recieved_data = s.recv(4096)
65      return recieved_data
66
67  def read_fixed_header(s, filename):
68      s.settimeout(1)
69      try:
```

```
   70           data = s.recv(8)
   71       except socket.timeout:
   72           print("ERROR: CONNECTION TIMEOUT")
   73           s.close()
   74           sys.exit()
   75
   76       MagicNo = (int).from_bytes(data[0:2], "big")
   77       Type = data[2]
   78       StatusCode = data[3]
   79       DataLength = (int).from_bytes(data[4:], "big")
   80
   81       FixedHeader = MagicNo + Type + StatusCode + DataLength
   82
   83       if MagicNo == 0x497E and Type == 2 and (StatusCode ==↵
        ↳    1 or StatusCode == 0):
   84           print("CONDITIONS ARE CORRECT")
   85           pass
   86       else:
   87           print("ERROR: FILE REQUEST IS ERRONEOUS")
   88           s.close()
   89
   90       if StatusCode == 0:
   91           print("ERROR: FILE DOES NOT EXIST ON SERVER SIDE")
   92           s.close()
   93           sys.exit()
   94       else:
   95           try:
   96               f = open(filename, "wb+") # wb+  =  create ↵
        ↳          write bytes
   97           except IOError:
   98               print("ERROR: FILE CANNOT BE OPENED FOR ↵
        ↳          WRITING")
   99               s.close()
  100               sys.exit()
  101
  102           DataLength_recieved = 0 #initialise
  103           while True:
  104               try:
  105                   f_data = s.recv(4096) #buffer
```

```
106                 except IOError:
107                     print("ERROR: FIXED HEADER IS ERRONEOUS, CONNECTION TIMEOUT")
108                     s.close()
109                     f.close()
110                     sys.exit()
111                 except socket.error:
112                     print("ERROR: FILE DATA CANNOT BE RECIEVED FROM SERVER")
113                     s.close()
114                     f.close()
115                     sys.exit()
116                 byte_array = bytearray(f_data)
117                 try:
118                     f.write(f_data)
119                 except IOError:
120                     print("ERROR WRITING TO FILE")
121                     s.close()
122                     f.close()
123                     sys.exit()
124                 if not f_data:
125                     break
126                 DataLength_recieved += len(f_data)
127             if DataLength_recieved != DataLength:
128                 print("ERROR: DATA BYTES VALID")
129                 s.close()
130                 sys.exit()
131 
132             print("FILE RECIEVED")
133             print("THE NUMBER OF BYTES RECIEVED IS: {}".format(DataLength_recieved))
134             f.close()
135             sys.exit()
136 
137 
138     def main():
139         less_than_three()
140         address = get_ip()
141         port = get_port()
```

```
  142       filename = name_of_file()
  143       s = create_socket()
  144       connect(s, address, port)
  145       fileRequest = file_request(filename)
  146       s.sendall(fileRequest)
  147       read_fixed_header(s, filename)
  148       recieve_data(s)
  149
  150   main()
```