Giselle Northy
Project 2 - Reliable Data Transmission (RDT)
Nov 2, 2020

Introduction

- In this coding project we started with skeleton code for an RDT layer that allows the transfer of string data through an unreliable channel in a simulated environment. Code for the unreliable channel, the Segment class, and also the calling main function, is provided. Essentially we had to build out and fill out missing functionality in the code.

How it works

- Used skeleton code template version 1 on Windows machine
- The skeleton code provided creates a client to send data to a server
- Both are running class **RDTLayer**.
- The client first manages sending and manages receiving. The server does the same.
- In the **manageSend** function, the client checks to see if any segments have timed out and will resend if needed.
- Then it will send segments broken up into **DATA LENGTH** which is 4 characters out that fit in the **FLOW CONTROL WINDOW** which is a constant of 15 characters.
- On **manageReceive,** it accepts all incoming messages from the channel
- If the sequence number is greater than the previous ACK number it will save the packet for processing later as long as the packet's checksum passes.
- If the sequence number matches the previous ACK number, it knows this is the next segment we are looking for.
    - It checks the packets Checksum to verify the data is not corrupt.
    - Saved packets are checked and added to the data received, if they are next in order.
- One ACK message is sent for an incoming set of messages as a cumulative ACK.
- If the incoming packet is an ACK message, all packets with sequence numbers less than the ACK are cleared for timeout.

How to run

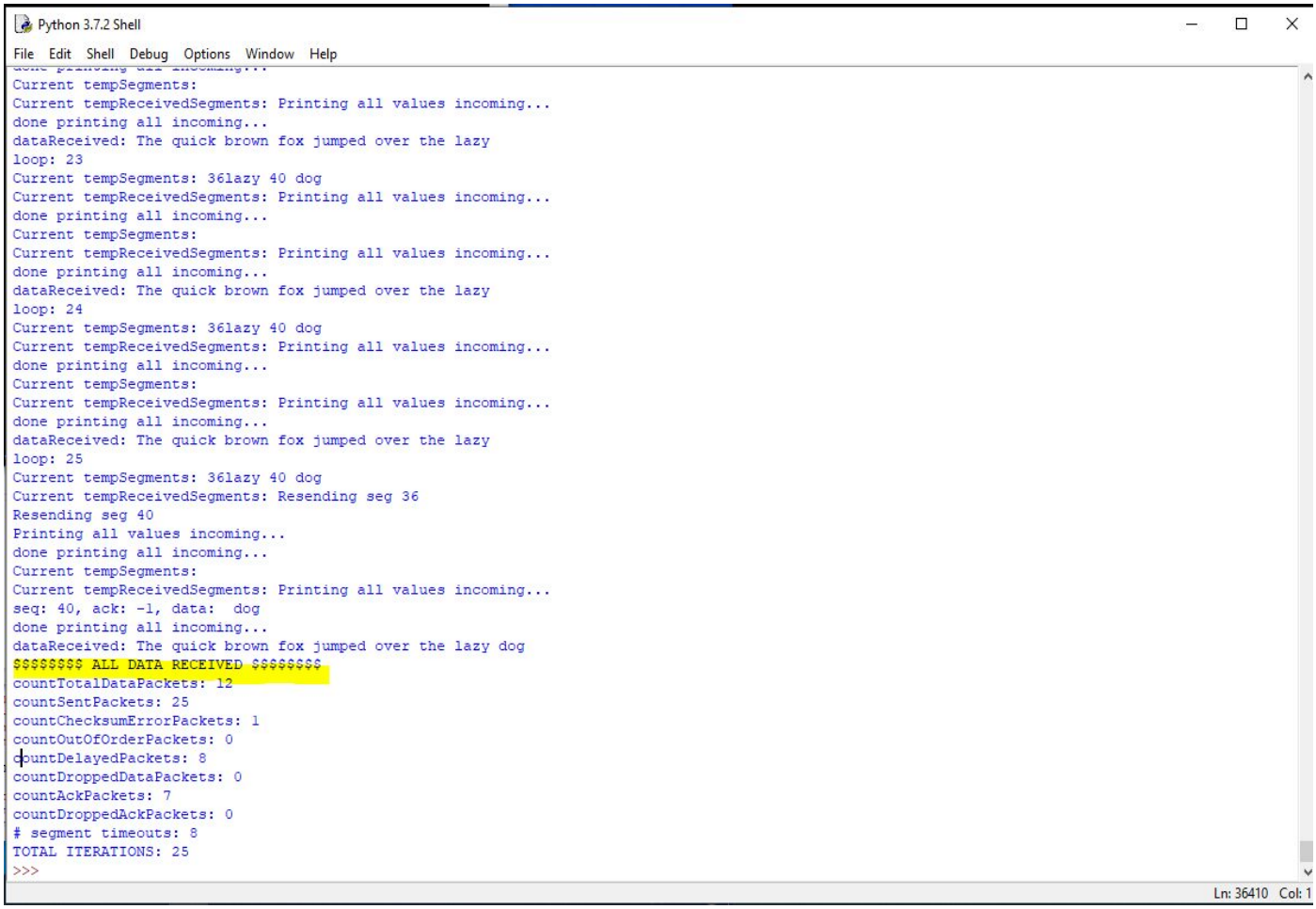
- Run the program from terminal Python from **rdt_main.py** for skeleton code version 1
- Example of the run command in a Windows terminal:
    - **py -3.7 rdt_main.py**

Comments

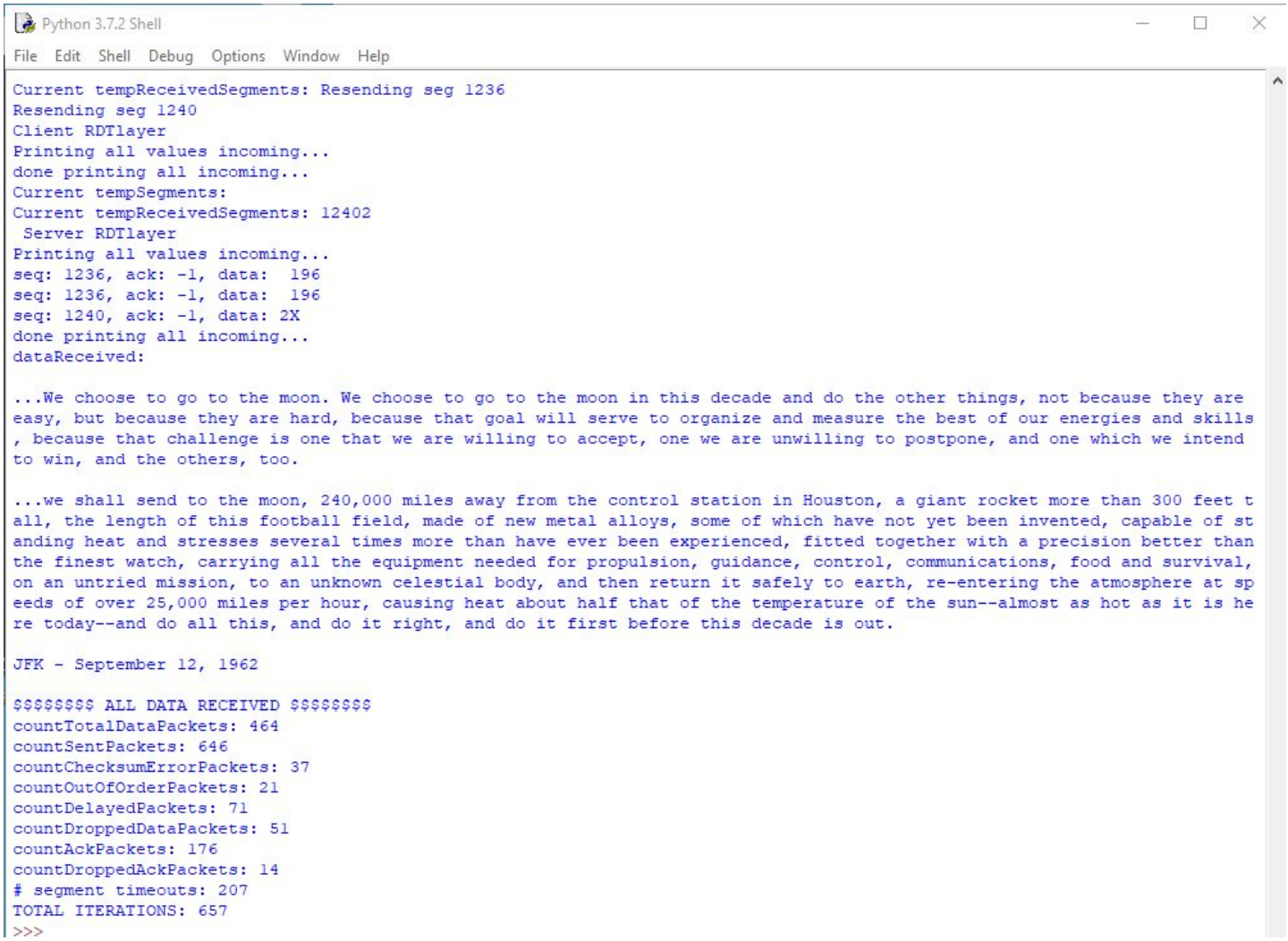
- I found this assignment to be very challenging. The concepts from the lectures related to RDT made sense, but finishing the Python skeleton code was hard. Partly I think this is because I have not programmed in Python much. The Piazza discussions were helpful and gave me some good ideas on how to get started.

Screenshots of running code

Example A) Running the shorter string “The quick brown fox jumped over the lazy dog”
Note: Included extra print statements to keep track of current segments



Example B) Running the “long” string “...We choose to go to the moon…”



Example C) Running in Command Line