CSE 3100: Assignment 6
Instructor: Kriti Bhargava

Q1. Pipeline command. (30 points)

Complete the `main` function in Q1.c to implement the following pipeline command:

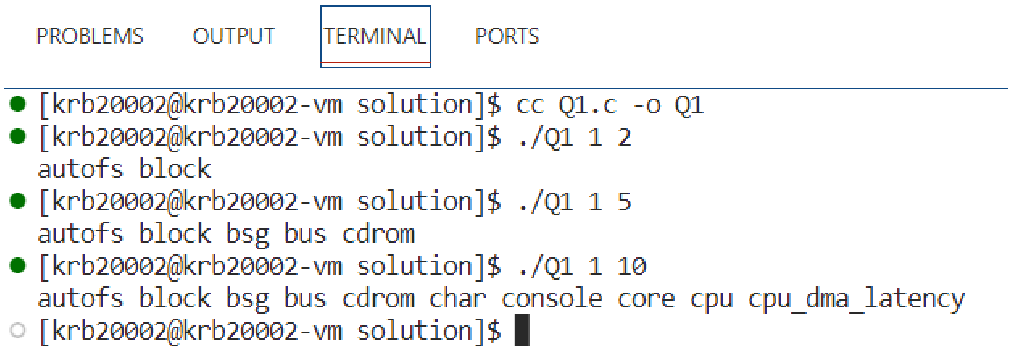
```
ls /dev | xargs | cut -d‘ ’ -f<a>-<b>
```

where `<a>` and `<b>` are positive integers specified as command line arguments of your program.

In this instance, the pipeline consists of three external commands, which can take zero or more arguments, chained together to form a single instruction. The pipeline has 3 stages:

1. The first stage outputs the list of files in `/dev`.
2. The second stage takes the output of stage 1 on its standard input, delimits it by blanks, and echos them on its standard output.
3. The third stage takes the output of stage 2 as its standard input and prints the selected fields (<a>-<b>) on the standard output delimited by a blank. Note that the pair of quotes after -d are not needed when passing arguments through execl.

Below are sample outputs from running the program on my system. Note: the output can vary on each machine. The length of the output, however, should be the same as the below examples.



```
● [krb20002@krb20002-vm solution]$ cc Q1.c -o Q1
● [krb20002@krb20002-vm solution]$ ./Q1 1 2
  autofs block
● [krb20002@krb20002-vm solution]$ ./Q1 1 5
  autofs block bsg bus cdrom
● [krb20002@krb20002-vm solution]$ ./Q1 1 10
  autofs block bsg bus cdrom char console core cpu cpu_dma_latency
○ [krb20002@krb20002-vm solution]$ █
```

Q2. A client-server application. (70 points)

Complete the code in Q2service.c, Q2client.c, and Q2server.c to implement a client-server application for SQLite. SQLite is a relational database management system that takes as input a query, consults a database, and returns with an appropriate response. The functionality to be implemented in each of the above programs is as follows:

1. *Q2server.c*: This is a server process that runs on the ‘localhost’ over TCP. The server is listening on two ports (taken as command line inputs from the user and stored in variables sid and srv) and must respond to inbound chat requests in the following way:

   a. Inbound requests on the first port (sid): run an instance of SQLite3 for each client, which reads and executes the stream of SQLite commands coming over the socket, and sends back the response. Note: the executable for SQLite is sqlite3. It takes as input a database file. You have been provided with foobar.db in the starter code.

   b. Inbound requests on the second port (srv): read commands from the socket and terminate the server if the command is '$die!'. Any other request should be met by producing the message (on a single line)

      ```
      bad command [<command that was sent>]\n
      ```

   c. reap all the dead children.

2. *Q2client.c*: This is a client process that must
   a. connect to the server process via the first port (sid) running on the hostname provided by the user. For testing purposes, your server (Q2server) is running on localhost.

   b. read SQLite commands from stdin and send them to the server. Note: you are provided with sample SQL queries in the scr1.sql file.

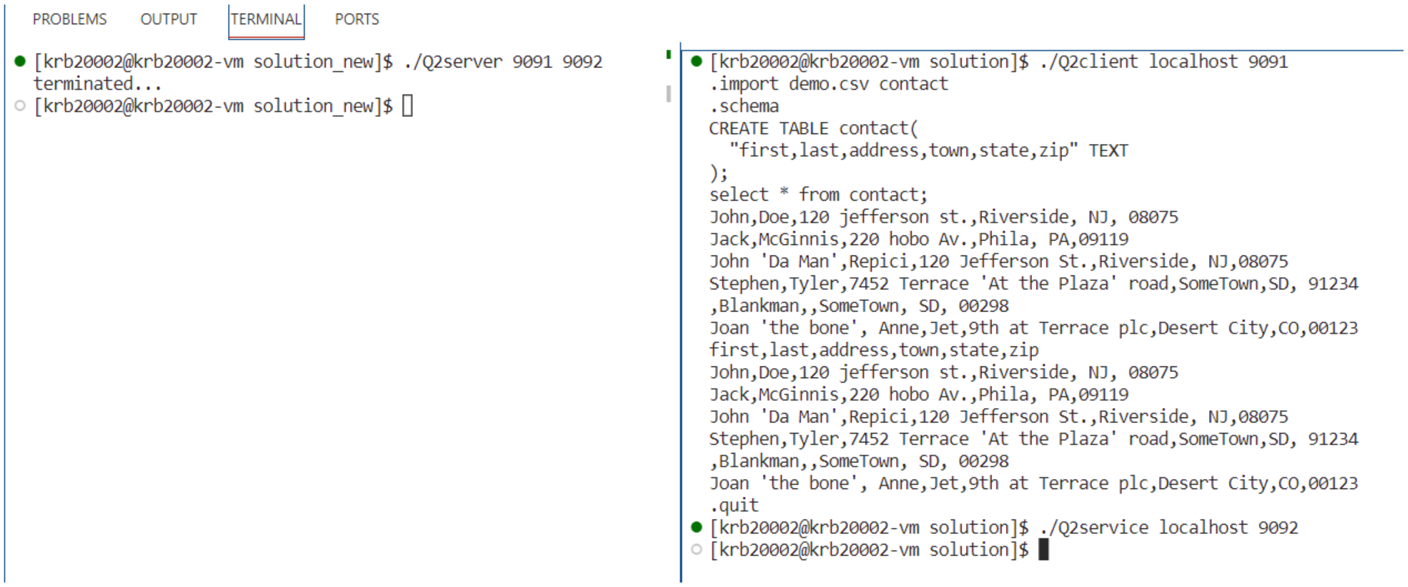
   c. read the responses from the server and print them out on stdout.

3. *Q2service.c*: This is a client process that must
   a. connect to the server process via the second port (srv) running on the hostname provided by the user. For testing purposes, your server (Q2server) is running on localhost.

   b. send the '`$die!`' command to terminate the server process.

You need to complete the `main` function in Q2server.c **(Q2.1),** Q2client.c **(Q2.2)** and Q2service.c **(Q2.3)** to implement the above functionalities. In case of Q2client.c, you are provided with the code to read a SQLite command from the stdin. In case of Q2server.c, you are provided with boilerplate code to create and bind a socket to a given address and listen for incoming TCP connect requests.

Below is a sample session of running the application.



PROBLEMS OUTPUT TERMINAL PORTS

```
● [krb20002@krb20002-vm solution_new]$ ./Q2server 9091 9092
  terminated...
○ [krb20002@krb20002-vm solution_new]$ 
```

```
● [krb20002@krb20002-vm solution]$ ./Q2client localhost 9091
  .import demo.csv contact
  .schema
  CREATE TABLE contact(
    "first,last,address,town,state,zip" TEXT
  );
  select * from contact;
  John,Doe,120 jefferson st.,Riverside, NJ, 08075
  Jack,McGinnis,220 hobo Av.,Phila, PA,09119
  John 'Da Man',Repici,120 Jefferson St.,Riverside, NJ,08075
  Stephen,Tyler,7452 Terrace 'At the Plaza' road,SomeTown,SD, 91234
  ,Blankman,,SomeTown, SD, 00298
  Joan 'the bone', Anne,Jet,9th at Terrace plc,Desert City,CO,00123
  first,last,address,town,state,zip
  John,Doe,120 jefferson st.,Riverside, NJ, 08075
  Jack,McGinnis,220 hobo Av.,Phila, PA,09119
  John 'Da Man',Repici,120 Jefferson St.,Riverside, NJ,08075
  Stephen,Tyler,7452 Terrace 'At the Plaza' road,SomeTown,SD, 91234
  ,Blankman,,SomeTown, SD, 00298
  Joan 'the bone', Anne,Jet,9th at Terrace plc,Desert City,CO,00123
  .quit
● [krb20002@krb20002-vm solution]$ ./Q2service localhost 9092
○ [krb20002@krb20002-vm solution]$ 
```