Daniel Steele
Lab 10

By default, our servers handle one connection at a time. Subsequent clients can connect and send messages but will not receive a repsonse. There are three types of concurrent servers - what are the names? Review the code below and describe which type of concurrent server is present.

```
        int main()
        {
         int sockfd;//to create socket
         int newsockfd;//to accept connection

         struct sockaddr_in serverAddress;//server receive on this address
         struct sockaddr_in clientAddress;//server sends to client on this
address

         int n;
         char msg[MAXSZ];
         int clientAddressLength;
         int pid;

         sockfd=socket(AF_INET,SOCK_STREAM,0);
         memset(&serverAddress,0,sizeof(serverAddress));
         serverAddress.sin_family=AF_INET;
         serverAddress.sin_addr.s_addr=htonl(INADDR_ANY);
         serverAddress.sin_port=htons(PORT);

         bind(sockfd,(struct sockaddr *)&serverAddress,
sizeof(serverAddress));

         listen(sockfd,5);

         while(1)
         {
          printf("\n*****server waiting for new client connection:*****\n");
          clientAddressLength=sizeof(clientAddress);
          newsockfd=accept(sockfd,(struct
sockaddr*)&clientAddress,&clientAddressLength);
          printf("connected to client:
%s\n",inet_ntoa(clientAddress.sin_addr));

          pid=fork();
          if(pid==0)//child process rec and send
          {
           while(1)
           {
            n=recv(newsockfd,msg,MAXSZ,0);
            if(n==0)
            {
             close(newsockfd);
             break;
            }
            msg[n]=0;
            send(newsockfd,msg,n,0);

            printf("Receive and set:%s\n",msg);
```

```
   }
  exit(0);
  }
  else
  {
   close(newsockfd);
  }
 }

 return 0;
}
```

The three types of concurrent servers are as follows:

- One Child Per Client (TCP/UDP) – Server uses fork to create a new child for each client that connects or sends a request
- One Thread Per Client (TCP) – Server creates a new thread (pthread_create) for each client that connects
- One Port Per Client (UDP) – Server binds a new port for each client that sends a request

The provided sample code in the prompt utilizes a One Child Per Client method. This can be determined by the fact that fork() is called after each accept and the new child goes on to process the connection.