/\*

\* isock\_server : listen on an internet socket ; fork ;

\* child does lookup ; replies down same socket

\* argv[1] is the name of the local datafile

\* PORT is defined in dict.h

\*/

#include <sys/types.h>

#include <sys/socket.h>

#include <netdb.h>

#include <netinet/in.h>

#include <errno.h>

#include <string.h>

#include <unistd.h>

#include "dict.h"

int main(int argc, char \*\*argv) {

static struct sockaddr\_in server;

int sd,cd,n;

Dictrec tryit;

struct sockaddr\_in client;

n = sizeof(client);

if (argc != 2) {

fprintf(stderr,"Usage : %s <datafile>\n",argv[0]);

exit(1);

}

/\* Create the socket.

\* Fill in code. \*/

if((sd=socket(AF\_INET,SOCK\_STREAM,0))==-1){

perror("socket");

exit(1);

}

/\* Initialize address.

\* Fill in code. \*/

memset(&server,'\0',sizeof(server));

server.sin\_family = AF\_INET;

server.sin\_port = htons(PORT);

/\* Name and activate the socket.

\* Fill in code. \*/

if(bind(sd,(struct sockaddr \*)&server,sizeof(server)) < 0){

perror("bind");

exit(1);

}

if(listen(sd,10) == -1 ){

perror("listen");

exit(1);

}

/\* main loop : accept connection; fork a child to have dialogue \*/

for (;;) {

/\* Wait for a connection.

\* Fill in code. \*/

cd =accept(sd,(struct sockaddr \*)&client,&n);

/\* Handle new client in a subprocess. \*/

switch (fork()) {

case -1 :

DIE("fork");

case 0 :

close (sd); /\* Rendezvous socket is for parent only. \*/

/\* Get next request.

\* Fill in code. \*/

while (recv(cd,&tryit,sizeof(Dictrec),0)) {

/\* Lookup the word , handling the different cases appropriately \*/

switch(lookup(&tryit,argv[1]) ) {

/\* Write response back to the client. \*/

case FOUND:

/\* Fill in code. \*/

send(cd,&tryit,sizeof(Dictrec),0);

break;

case NOTFOUND:

/\* Fill in code. \*/

strcpy(tryit.text,"XXXX");

send(cd,&tryit,sizeof(Dictrec),0);

break;

case UNAVAIL:

DIE(argv[1]);

} /\* end lookup switch \*/

} /\* end of client dialog \*/

exit(0); /\* child does not want to be a parent \*/

default :

close(cd);

break;

} /\* end fork switch \*/

} /\* end forever loop \*/

} /\* end main \*/