 1.Bit Stuffing Program in C

#include<stdio.h>

#include<string.h>

int main()

{

int a[20],b[30],i,j,k,count,n;

printf("Enter frame size (Example: 8):");

scanf("%d",&n);

printf("Enter the frame in the form of 0 and 1 :");

for(i=0; i<n; i++)

scanf("%d",&a[i]);

i=0;

count=1;

j=0;

while(i<n)

{

if(a[i]==1)

{

b[j]=a[i];

for(k=i+1; a[k]==1 && k<n && count<5; k++)

{

j++;

b[j]=a[k];

count++;

if(count==5)

{

j++;

b[j]=0;

}

i=k;

}

}

else

{

b[j]=a[i];

}

i++;

j++;

}

printf("After Bit Stuffing :");

for(i=0; i<j; i++)

printf("%d",b[i]);

return 0;

}

2. TCP Socket in C

#include <stdio.h>

#include <netdb.h>

#include <netinet/in.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#include <sys/types.h>

#define MAX 80

#define PORT 8080

#define SA struct sockaddr

void func(int connfd)

{

char buff[MAX];

int n;

for (;;) {

bzero(buff, MAX);

read(connfd, buff, sizeof(buff));

printf("From client: %s\t To client : ", buff);

bzero(buff, MAX);

n = 0;

while ((buff[n++] = getchar()) != '\n')

;

write(connfd, buff, sizeof(buff));

if (strncmp("exit", buff, 4) == 0) {

printf("Server Exit...\n");

break;

}

}

}

int main()

{

int sockfd, connfd, len;

struct sockaddr\_in servaddr, cli;

sockfd = socket(AF\_INET, SOCK\_STREAM, 0);

if (sockfd == -1) {

printf("socket creation failed...\n");

exit(0);

}

else

printf("Socket successfully created..\n");

bzero(&servaddr, sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_addr.s\_addr = htonl(INADDR\_ANY);

servaddr.sin\_port = htons(PORT);

if ((bind(sockfd, (SA\*)&servaddr, sizeof(servaddr))) != 0) {

printf("socket bind failed...\n");

exit(0);

}

else

printf("Socket successfully binded..\n");

if ((listen(sockfd, 5)) != 0) {

printf("Listen failed...\n");

exit(0);

}

else

printf("Server listening..\n");

len = sizeof(cli);

connfd = accept(sockfd, (SA\*)&cli, &len);

if (connfd < 0) {

printf("server accept failed...\n");

exit(0);

}

else

printf("server accept the client...\n");

func(connfd);

close(sockfd);

}

#include <netdb.h>

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <sys/socket.h>

#define MAX 80

#define PORT 8080

#define SA struct sockaddr

void func(int sockfd)

{

char buff[MAX];

int n;

for (;;) {

bzero(buff, sizeof(buff));

printf("Enter the string : ");

n = 0;

while ((buff[n++] = getchar()) != '\n')

;

write(sockfd, buff, sizeof(buff));

bzero(buff, sizeof(buff));

read(sockfd, buff, sizeof(buff));

printf("From Server : %s", buff);

if ((strncmp(buff, "exit", 4)) == 0) {

printf("Client Exit...\n");

break;

}

}

}

int main()

{

int sockfd, connfd;

struct sockaddr\_in servaddr, cli;

sockfd = socket(AF\_INET, SOCK\_STREAM, 0);

if (sockfd == -1) {

printf("socket creation failed...\n");

exit(0);

}

else

printf("Socket successfully created..\n");

bzero(&servaddr, sizeof(servaddr));

servaddr.sin\_family = AF\_INET;

servaddr.sin\_addr.s\_addr = inet\_addr("127.0.0.1");

servaddr.sin\_port = htons(PORT);

if (connect(sockfd, (SA\*)&servaddr, sizeof(servaddr)) != 0) {

printf("connection with the server failed...\n");

exit(0);

}

else

printf("connected to the server..\n");

func(sockfd);

close(sockfd);}

3. **UDP Socket in C**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <string.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <arpa/inet.h>

#include <netinet/in.h>

#define PORT 8080

#define MAXLINE 1024

int main() {

int sockfd;

char buffer[MAXLINE];

char \*hello = "Hello from server";

struct sockaddr\_in servaddr, cliaddr;

if ( (sockfd = socket(AF\_INET, SOCK\_DGRAM, 0)) < 0 ) {

perror("socket creation failed");

exit(EXIT\_FAILURE);

}

memset(&servaddr, 0, sizeof(servaddr));

memset(&cliaddr, 0, sizeof(cliaddr));

servaddr.sin\_family = AF\_INET; // IPv4

servaddr.sin\_addr.s\_addr = INADDR\_ANY;

servaddr.sin\_port = htons(PORT);

if ( bind(sockfd, (const struct sockaddr \*)&servaddr,

sizeof(servaddr)) < 0 )

4. **CRC program in C**

#include <stdio.h>

#include <conio.h>

#include <string.h>

void main() {

int i,j,keylen,msglen;

char input[100], key[30],temp[30],quot[100],rem[30],key1[30];

clrscr();

printf("Enter Data: ");

gets(input);

printf("Enter Key: ");

gets(key);

keylen=strlen(key);

msglen=strlen(input);

strcpy(key1,key);

for (i=0;i<keylen-1;i++) {

input[msglen+i]='0';

}

for (i=0;i<keylen;i++)

temp[i]=input[i];

for (i=0;i<msglen;i++) {

quot[i]=temp[0];

if(quot[i]=='0')

for (j=0;j<keylen;j++)

key[j]='0'; else

for (j=0;j<keylen;j++)

key[j]=key1[j];

for (j=keylen-1;j>0;j--) {

if(temp[j]==key[j])

rem[j-1]='0'; else

rem[j-1]='1';

}

rem[keylen-1]=input[i+keylen];

strcpy(temp,rem);

}

strcpy(rem,temp);

printf("\nQuotient is ");

for (i=0;i<msglen;i++)

printf("%c",quot[i]);

printf("\nRemainder is ");

for (i=0;i<keylen-1;i++)

printf("%c",rem[i]);

printf("\nFinal data is: ");

for (i=0;i<msglen;i++)

printf("%c",input[i]);

for (i=0;i<keylen-1;i++)

printf("%c",rem[i]);

getch(); }

5. **To determine IP Address & hostname of Local Computer in C.**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <errno.h>

#include <netdb.h>

#include <sys/types.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

// Returns hostname for the local computer

void checkHostName(int hostname)

{

if (hostname == -1)

{

perror("gethostname");

exit(1);

}

}

// Returns host information corresponding to host name

void checkHostEntry(struct hostent \* hostentry)

{

if (hostentry == NULL)

{

perror("gethostbyname");

exit(1);

}

}

// Converts space-delimited IPv4 addresses

// to dotted-decimal format

void checkIPbuffer(char \*IPbuffer)

{

if (NULL == IPbuffer)

{

perror("inet\_ntoa");

exit(1);

}

}

int main()

{

char hostbuffer[256];

char \*IPbuffer;

struct hostent \*host\_entry;

int hostname;

hostname = gethostname(hostbuffer, sizeof(hostbuffer));

checkHostName(hostname);

host\_entry = gethostbyname(hostbuffer);

checkHostEntry(host\_entry);

IPbuffer = inet\_ntoa(((struct in\_addr)

host\_entry->h\_addr\_list[0]));

printf("Hostname: %s\n", hostbuffer);

printf("Host IP: %s", IPbuffer);

return 0;

}