**PROGRAM CODE:**

#include<stdio.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<stdlib.h>

#include<string.h>

#include<unistd.h>

#include<arpa/inet.h>

#include<fcntl.h>

#include<stdbool.h>

#include<math.h>

#define MAXLINE 1024

int binary(int num)

{

int bin=0,r;

int i=0;

while(num>0)

{

r=num%2;

bin+=r\*pow(10,i);

num/=2;

i++;

}

return bin;

}

int ispresent(int num,int pos)

{

int rem;

for(int i=0;i<pos;i++)

{

rem=num%10;

num=num/10;

}

if(rem==1)

return 1;

else

return 0;

}

int isapower2(int n)

{

if(ceil(log2(n)) == floor(log2(n)))

return 1;

else

return 0;

}

int main()

{

int sockfd,newfd,n=0,arr[30],count=0,bin;// n-no.of bits

char buff[MAXLINE],buffer[MAXLINE],data\_t[40]; //char buffer1[MAXLINE],token[100];

int i,j,r,total,nob,rem,dig,pos;

long data;

struct sockaddr\_in servaddr,cliaddr;

// Creating socket file descriptor

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

{

perror("socket creation failed");

exit(EXIT\_FAILURE);

}

bzero(&servaddr,sizeof(servaddr));

// Filling server information

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

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

servaddr.sin\_port = htons(8080);

// Bind the socket with the server address

if ( bind(sockfd, (const struct sockaddr \*)&servaddr,sizeof(servaddr)) < 0 )

{

perror("bind failed");

exit(EXIT\_FAILURE);

}

int len,m;

listen(sockfd,2);

len=sizeof(cliaddr);

printf("Enter the data:");

scanf("%ld",&data);

int temp=data;

while(temp>0)

{

temp=temp/10;

n++;

}

r=0;

while(pow((double)2,(double)r)<(n+r+1))

{

r++;

}

printf("\nNo.of redundant bits:%d\n",r);

total=n+r;

for(i=1;i<=total;i++)

{

dig=data%10;

if(isapower2(i)==0)

{

arr[total-i]=dig;

data/=10;

}

else

arr[total-i]=0;

}

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

{

for(j=1;j<=total;j++)

{

bin=binary(j);

if(ispresent(bin,i+1))

count+=arr[total-j];

}

if(count%2==0)

arr[total-(int)pow(2,i)]=0;

else

arr[total-(int)pow(2,i)]=1;

count=0;

}

printf("\nData with redundant bits:");

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

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

printf("\n\nEnter the position where error has to be made:");

scanf("%d",&pos);

if(arr[total-pos]==0)

arr[total-pos]=1;

else

arr[total-pos]=0;

printf("\n");

int k=0;

long num=0;

for(i=total-1;i>=0;i--)

{

num+=pow(10,k)\*arr[i];

k++;

}

sprintf(data\_t, "%ld", num);

printf("Data transmitted is %s\n",data\_t);

newfd=accept(sockfd,(struct sockaddr\*)&cliaddr,&len);

m=write(newfd,data\_t,sizeof(data\_t));

}

#include<stdio.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

#include<string.h>

#include<unistd.h>

#include<arpa/inet.h>

#include<stdlib.h>

#include<math.h>

#define PORT 8080

#define MAXLINE 1024

int countbits(long num)

{

int count=0;

while(num>0)

{

num=num/10;

count++;

}

return count;

}

int binary(int num)

{

int bin=0,r;

int i=0;

while(num>0)

{

r=num%2;

bin+=r\*pow(10,i);

num/=2;

i++;

}

return bin;

}

int decimal(int num)

{

int rem,i=0,result;

while(num>0)

{

rem=num%10;

result+=pow(2,i)\*rem;

num/=10;

i++;

}

return result;

}

int ispresent(int num,int pos)

{

int rem;

for(int i=0;i<pos;i++)

{

rem=num%10;

num=num/10;

}

if(rem==1)

return 1;

else

return 0;

}

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

{

long num;

int sockfd,total,i,rem,arr[20],count=0,r=0,result=0,bin,j,newarr[20],finalarr[20];

char buffer1[40];

struct sockaddr\_in servaddr;

// Creating socket file descriptor

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

perror("socket creation failed");

exit(EXIT\_FAILURE);

}

bzero(&servaddr,sizeof(servaddr));

// Filling server information

servaddr.sin\_family = AF\_INET;

servaddr.sin\_port = htons(PORT);

servaddr.sin\_addr.s\_addr = inet\_addr(argv[1]);

int n, len;

connect(sockfd,(struct sockaddr\*)&servaddr,sizeof(servaddr));

n=read(sockfd,buffer1,sizeof(buffer1));

// printf("%s\n",buffer1);

num=atol(buffer1);

total=countbits(num);

//printf("total :%d\n",total);

printf("Received data:%lu\n",num);

i=1;

while(num>0)

{

rem=num%10;

arr[total-i]=rem;

num/=10;

i++;

}

for(i=1;i<=total;i++)

if(ceil(log2(i)) == floor(log2(i)))

r++; //no. of redundant bits

// printf("r is %d\n",r);

int k=0;

for(i=0;i<4;i++)

{

for(j=1;j<=total;j++)

{

bin=binary(j);

if(ispresent(bin,i+1))

count+=arr[total-j];

}

// printf("count:%d\n",count);

if(count%2==0)

result+=pow(10,k)\*0;

else

result+=pow(10,k)\*1;

k++;

count=0;

}

int error=decimal(result);

printf("\nError bit in binary:%d\n",result);

printf("\nError in bit-%d\n",error);

if(arr[total-error]==0)

arr[total-error]=1;

else

arr[total-error]=0;

k=0;

printf("\nData after error correction:");

for(i=total-1;i>=0;i--)

{

newarr[k]=arr[i];

k++;

}

int x=0;

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

{

if(ceil(log2(i+1)) != floor(log2(i+1)))

{

finalarr[x]=newarr[i];

x++;

}

}

for(i=x-1;i>=0;i--)

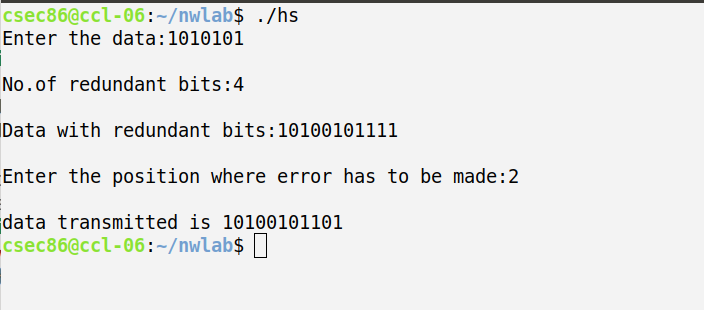
printf("%d",finalarr[i]);

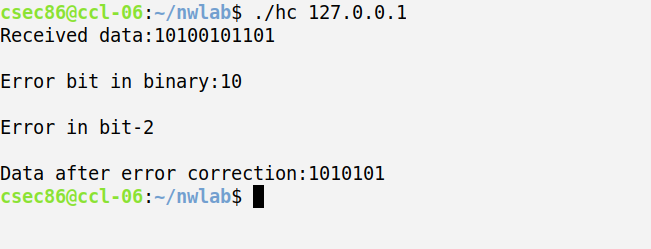
printf("\n");

return 0;

}

**OUTPUT:**

****

****

**PROGRAM CODE:**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<sys/socket.h>

#include<netdb.h>

#include<arpa/inet.h>

#include<sys/types.h>

#include<unistd.h>

#include<fcntl.h>

int main(int argc, char \*argv[])

{

int socket\_desc, i, bytes\_read;

char server\_reply[1024], ip[100], request[100];

char \*hostname = argv[1];

struct sockaddr\_in server;

struct hostent \*he;

struct in\_addr \*\*addr\_list;

FILE \*fp;

if ((he = gethostbyname(hostname)) == NULL) {

//gethostbyname failed

herror("gethostbyname\n");

return 1;

}

addr\_list = (struct in\_addr \*\*) he->h\_addr\_list;

for(i = 0; addr\_list[i] != NULL; i++) {

//Return the first one;

strcpy(ip , inet\_ntoa(\*addr\_list[i]) );

}

//Create socket

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

if (socket\_desc == -1) {

printf("Could not create socket!\n");

}

server.sin\_addr.s\_addr = inet\_addr(ip);

server.sin\_family = AF\_INET;

server.sin\_port = htons(80);

//Connect to remote server

if (connect(socket\_desc , (struct sockaddr \*)&server , sizeof(server)) < 0)

{

printf("connect error!\n");

return 1;

}

printf("Connected...\n");

//Send some data

snprintf(request, 99, "GET / HTTP/1.1\r\n"

"Host: %s\r\n"

"\r\n\r\n", hostname);

if (send(socket\_desc, request, strlen(request), 0) < 0) {

puts("Send failed!\n");

return 1;

}

puts("Data Sent...\n");

//Receive a reply from the server

fp = fopen("/home/csec86/Desktop/ouput.html", "w+");

//printf("\nhi\n");

while (bytes\_read = read(socket\_desc, server\_reply, sizeof(server\_reply)) >

0)

{

//printf("\n%dhi\n",bytes\_read);

fputs(server\_reply, fp);

//printf("\n%dhelo\n",bytes\_read);

memset(server\_reply, 0, sizeof(server\_reply));

//printf("\n%dhello\n",bytes\_read);

}

//printf("\nhi\n");

do {

bytes\_read = read(socket\_desc, server\_reply, sizeof(server\_reply));

fputs(server\_reply, fp);

memset(server\_reply, 0, sizeof(server\_reply));

} while (bytes\_read > 0);

printf("reply received...\n");

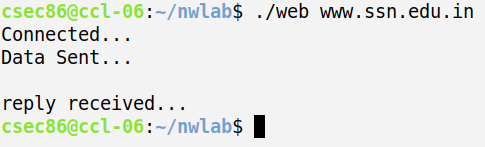
fclose(fp);

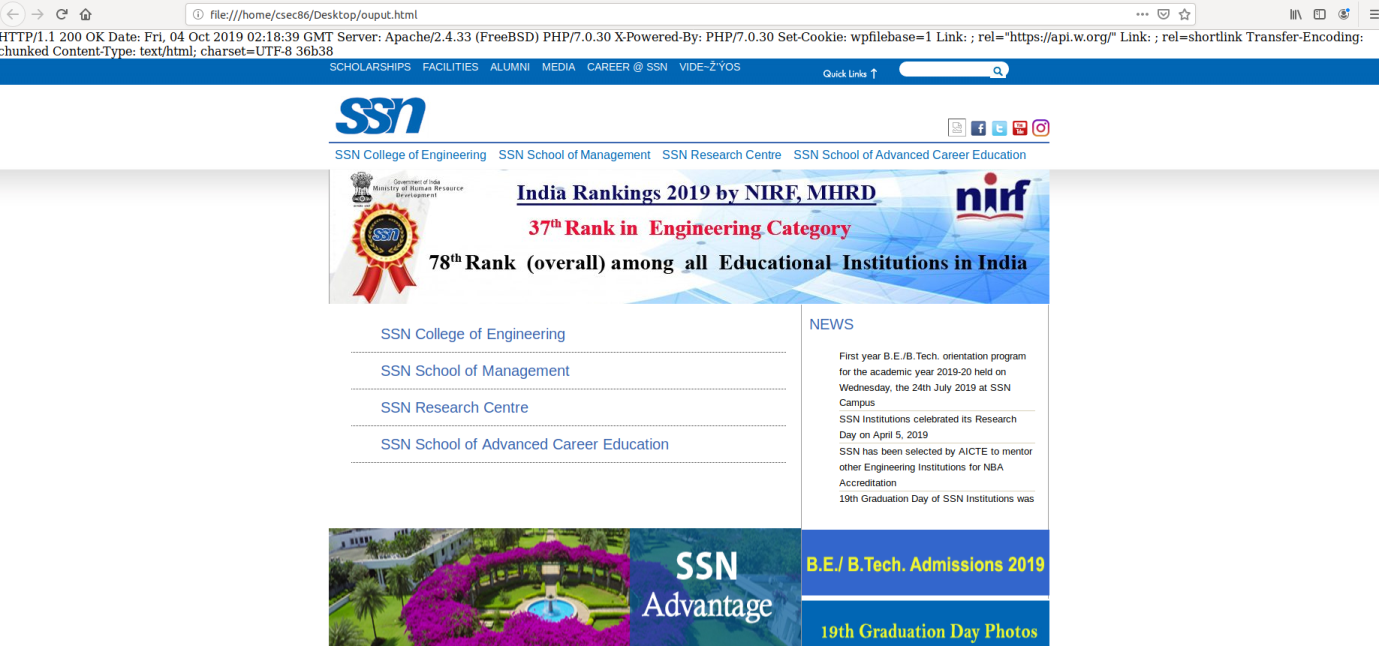
close(socket\_desc);

return 0;

}

**OUTPUT:**

****

****