Checksum

Sender

```
#include<stdio.h>
#include<string.h>

int main()
{
   char data[100];
   char ndata[100];
   int n;
   printf("Enter the Data\n");
   scanf("%s", data);
   printf("Enter the block size\n");
   scanf("%d", &n);
   int len=strlen(data);
   if(len%n!=0)
   {
      int l=n-len%n;
      for(int i=0;i<l;i++)
      {
         ndata[i]='0';
      }
      for(int i=0;i<len;i++)
      {
         ndata[l+i]=data[i];
      }
      ndata[l+len]='\0';
   }
   else
   {
    strcpy(ndata, data);
}
char sum[n+1];
sum[n]='\0';
for(int i=0;i<n;i++)
{
    sum[i]=ndata[i];
}
char ca='0';
for(int i=n;i<strlen(ndata);i+=n)
{
    for(int j=n-1;j>=0;j--)
    {
        int n1=sum[j]=='0'?0:1;
        int n2=ndata[i+j]=='0'?0:1;
        int n3=ca=='0'?0:1;
        int d=n1+n2+n3;
        if(d==0)
        {
            sum[j]='0';
            ca='0';
        }
        else if(d==1)
        {
            sum[j]='1';
            ca='0';
        }
        else if(d==2)
        {
            sum[j]='0';
            ca='1';
        }
        else
        {
            sum[j]='1';
            ca='1';
        }
    }
    if(ca=='1')
    {
        char nsum[n];
        ca='0';
        for(int k=0;k<n-1;k++)
            nsum[k]='0';
        nsum[n-1]='1';
        char ca1='0';
        for(int j=n-1;j>=0;j--)
        {
            int n1=sum[j]=='0'?0:1;
            int n2=nsum[j]=='0'?0:1;
            int n3=ca1=='0'?0:1;
            int d=n1+n2+n3;
            if(d==0)
            {
                sum[j]='0';
                ca1='0';
            }
            else if(d==1)
            {
                sum[j]='1';
                ca1='0';
            }
```

```
            else if(d==2)
            {
                sum[j]='0';
                ca1='1';
            }
            else
            {
                sum[j]='1';
                ca1='1';
            }
        }

    }

  }
  for(int i=0;i<n;i++)
      sum[i]=sum[i]=='0'?'1':'0';
  strcat(data, sum);
  printf("Generated Code- %s", data);

}
```

Receiver

```
//@uthor-evilgenius(Swanmoy)
#include<stdio.h>
#include<string.h>

int main()
{
  char data[100];
  char ndata[100];
  int n;
```

```
printf("Enter the Data\n");
scanf("%s", data);
printf("Enter the block size\n");
scanf("%d", &n);
int len=strlen(data);
if(len%n!=0)
{
   int l=n-len%n;
   for(int i=0;i<l;i++)
   {
      ndata[i]='0';
   }
   for(int i=0;i<len;i++)
   {
      ndata[l+i]=data[i];
   }
   ndata[l+len]='\0';
}
else
{
   strcpy(ndata, data);
}
char sum[n+1];
sum[n]='\0';
for(int i=0;i<n;i++)
{
   sum[i]=ndata[i];
}
char ca='0';
for(int i=n;i<strlen(ndata);i+=n)
{
```

```
for(int j=n-1;j>=0;j--)
{
   int n1=sum[j]=='0'?0:1;
   int n2=ndata[i+j]=='0'?0:1;
   int n3=ca=='0'?0:1;
   int d=n1+n2+n3;
   if(d==0)
   {
      sum[j]='0';
      ca='0';
   }
   else if(d==1)
   {
      sum[j]='1';
      ca='0';
   }
   else if(d==2)
   {
      sum[j]='0';
      ca='1';
   }
   else
   {
      sum[j]='1';
      ca='1';
   }
}
if(ca=='1')
{
   char nsum[n];
   ca='0';
```

```
for(int k=0;k<n-1;k++)
   nsum[k]='0';
nsum[n-1]='1';
char ca1='0';
for(int j=n-1;j>=0;j--)
{
   int n1=sum[j]=='0'?0:1;
   int n2=nsum[j]=='0'?0:1;
   int n3=ca1=='0'?0:1;
   int d=n1+n2+n3;
   if(d==0)
   {
      sum[j]='0';
      ca1='0';
   }
   else if(d==1)
   {
      sum[j]='1';
      ca1='0';
   }
   else if(d==2)
   {
      sum[j]='0';
      ca1='1';
   }
   else
   {
      sum[j]='1';
      ca1='1';
   }
}
```

```
        }


    }
    int chk=0;
    for(int i=0;i<n;i++)
    {
        if(sum[i]=='0')
        {
            chk=1;
            break;
        }
    }
    if(chk==1)
        printf("The Received Data %s is incorrect\n", data);
    else
        printf("The Received Data %s is correct\n", data);


}
```

Cyclic Redundancy Code

Sender

```
#include<stdio.h>
#include<string.h>
int main()
{
    char data[100];
    char div[20];
    printf("Enter the data\n");
    scanf("%s", data);
    printf("Enter the divisor\n");
```

```
scanf("%s", div);
char code[100];
strcpy(code, data);
int c=strlen(data);
int l2=strlen(div);
int i, j;
for(i=0;i<l2-1;i++)
   strcat(data, "0");
int l1=strlen(data);
char rem[l2];
for(i=0;i<l2-1;i++)
   rem[i]=data[i];
rem[i]='\0';
for(i=l2-1;i<l1;i++)
{
   rem[l2-1]=data[i];
   char chf=rem[0];
      for(j=1;j<l2;j++)
      {
         if(chf=='0')
         {
            rem[j-1]=rem[j];
         }
         else
         {
            if(rem[j]==div[j])
               rem[j-1]='0';
            else
               rem[j-1]='1';
         }
      }
```

```
    }

    for(i=0;i<l2-1;i++)
    {
        code[c++]=rem[i];
    }
    code[c]='\0';
    printf("Generated Code- %s\n", code);
}
```

Receiver

```
#include<stdio.h>
#include<string.h>

int main()
{
    char data[100];
    char div[20];
    printf("Enter the data\n");
    scanf("%s", data);
    printf("Enter the divisor\n");
    scanf("%s", div);
    char code[100];
    strcpy(code, data);
    int c=strlen(data);
    int l2=strlen(div);
    int i, j;
    for(i=0;i<l2-1;i++)
        strcat(data, "0");
    int l1=strlen(data);
    char rem[l2];
```

```
for(i=0;i<l2-1;i++)
    rem[i]=data[i];
rem[i]='\0';
for(i=l2-1;i<l1;i++)
{
     rem[l2-1]=data[i];
    char chf=rem[0];
        for(j=1;j<l2;j++)
        {
            if(chf=='0')
            {
                rem[j-1]=rem[j];
            }
            else
            {
                if(rem[j]==div[j])
                    rem[j-1]='0';
                else
                    rem[j-1]='1';
            }
        }
}
int chkco=1;
for(i=0;i<l2;i++)
{
    if(rem[i]!='0')
    {
        chkco=0;
        break;
    }
}
```

```
  if(chkco)
    printf("The data %s is correct\n", code);
  else
    printf("The data %s is corrupted\n", code);
}
```

Hamming Code

Sender

```
//@uthor-evilgenius(Swanmoy)
#include<stdio.h>
#include<string.h>
#include<math.h>
int main()
{
  char data[100];
  printf("Enter the data\n");
  scanf("%s", data);
  int m=strlen(data);
  int r=1;
  while(pow(2, r)<m+r+1)
  {
    r++;
  }
  int l1=m+r;
  char code[m+r+1];
  int c=0;
  int k=0;
  for(int i=l1-1;i>=0;i--)
  {
    if(l1-i==pow(2, c))
      {
```

```
                code[i]='0';
                c++;
            }
            else
             {
                 code[i]=data[m-k-1];
                 k++;
             }
    }
    code[l1+1]='\0';
    c=0;
    while(pow(2, c)<=l1)
    {
        int cnt=0;
        int a=pow(2, c);
        for(int i=1;a*i<=l1;i+=2)
        {
            for(int j=a*i;j<a*(i+1)&&j<=l1;j++)
            {
                if(code[l1-j]=='1')
                    cnt++;
            }
        }
        if(cnt%2!=0)
            code[l1-a]='1';
        c++;

    }
    code[l1+1]='\0';
    printf("Generated Code- %s", code);
}
```

Receiver

```
//@uthor-evilgenius(Swanmoy)
#include<stdio.h>
#include<string.h>
#include<math.h>
int main()
{
    char data[100];
    printf("Enter the data\n");
    scanf("%s", data);
    int m=strlen(data);
    int ham[100];
    int r=0;
    while(pow(2, r)<=m)
    {
        int a=pow(2, r);
        int cnt=0;
        for(int i=1;a*i<=m;i+=2)
        {
            for(int j=a*i;j<a*(i+1)&&j<=m;j++)
            {
                if(data[m-j]=='1')
                    cnt++;
            }
        }
        ham[r]=cnt%2==0?0:1;
        r++;
    }
    int sum=0;
    for(int i=0;i<r;i++)
    {
        sum+=pow(2, i)*ham[i];
    }
    if(sum==0)
        printf("The code %s is correct\n", data);
    else
    {
        data[m-sum]=data[m-sum]=='0'?'1':'0';
        printf("The data was corrupted and the correct data is %s", data);
    }
}
```

Simple Server

Server

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <sys/types.h>
#include <netinet/in.h>
#include <sys/socket.h>
#include <arpa/inet.h>
int main(int x, char* argv[]) {
char buf[100],data[100];
memset(buf,'\0',100);
struct sockaddr_in server, client;
int s_check, c_check;
s_check = socket(AF_INET, SOCK_STREAM, 0);
server.sin_family = AF_INET;
server.sin_addr.s_addr = INADDR_ANY;
server.sin_port = htons(atoi(argv[1]));
bind(s_check,(struct sockaddr *)&server, sizeof(server));
listen(s_check, 2);
while(1) {
int size = sizeof(client);
c_check = accept(s_check, (struct sockaddr*)&client, &size);
memset(buf,'\0',100);
printf("\n entr data");
gets(data);
strcpy(buf,data);
send(c_check, buf, 100, 0);
printf("\nClient IP address is: %s\n", inet_ntoa(client.sin_addr));
printf("\nLocal port is: %d\n", (int) ntohs(client.sin_port));
recv(c_check, buf, 100, 0);
printf("\nRecieved data is : %s\n", buf);
close(c_check);
}
close(s_check);
return(0);
}
```

Client

```
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/ioctl.h>
#include <netinet/in.h>
#include <net/if.h>
#include <unistd.h>
#include <arpa/inet.h>
int main(int x, char * argv[]) {
struct sockaddr_in client;
int c_check;
char buf[100],data[100];

memset(buf, '\0', 100);

memset(data, '\0', 100);

c_check = socket(AF_INET, SOCK_STREAM, 0);

client.sin_family = AF_INET;

client.sin_addr.s_addr = inet_addr(argv[1]);

client.sin_port = htons(atoi(argv[2]));

connect(c_check, (struct sockaddr*)&client, sizeof(client));

recv(c_check, buf, 100, 0);

printf("\n codeword recv from sender is %s \n",buf);

printf("Enter data");

gets(buf);

send(c_check, buf, 100, 0);

close(c_check);

return(0);

}
```

Server Chatbot

Server

```
#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <sys/types.h>

#include <netinet/in.h>

#include <sys/socket.h>

#include <arpa/inet.h>

int main(int x, char* argv[]) {

char buf[100],data[100];

memset(buf,'\0',100);

struct sockaddr_in server, client;

int s_check, c_check;
s_check = socket(AF_INET, SOCK_STREAM, 0);
server.sin_family = AF_INET;
server.sin_addr.s_addr = INADDR_ANY;
server.sin_port = htons(atoi(argv[1]));
bind(s_check,(struct sockaddr *)&server, sizeof(server));
listen(s_check, 2);
int size = sizeof(client);
while(1)
{
c_check = accept(s_check, (struct sockaddr*)&client, &size);
memset(buf,'\0',100);
while(1) {
printf("\n enter data: ");
gets(data);
strcpy(buf,data);
send(c_check, buf, 100, 0);
if(strcmp(buf, "BYE")==0)
break;
printf("\nClient IP address is: %s\n", inet_ntoa(client.sin_addr));
printf("\nLocal port is: %d\n", (int) ntohs(client.sin_port));
recv(c_check, buf, 100, 0);
printf("\nRecieved data is : %s\n", buf);
if(strcmp(buf, "BYE")==0)
break;
}
close(c_check);
}
close(s_check);


return(0);
}
```

Client

```
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/ioctl.h>
#include <netinet/in.h>
#include <net/if.h>
#include <unistd.h>
#include <arpa/inet.h>
int main(int x, char * argv[]) {
struct sockaddr_in client;
int c_check;
char buf[100],data[100];
memset(buf, '\0', 100);
memset(data, '\0', 100);
c_check = socket(AF_INET, SOCK_STREAM, 0);
client.sin_family = AF_INET;
client.sin_addr.s_addr = inet_addr(argv[1]);
client.sin_port = htons(atoi(argv[2]));
connect(c_check, (struct sockaddr*)&client, sizeof(client));
while(1)
{
recv(c_check, buf, 100, 0);
printf("\n codeword recv from sender is %s \n",buf);
if(strcmp(buf, "BYE")==0)
break;
printf("Enter data");
gets(buf);
send(c_check, buf, 100, 0);
if(strcmp(buf, "BYE")==0)
break;
}
close(c_check);
return(0);
}
```

Server with Child Process(Multi Client Communication)

Server

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <sys/types.h>
#include<unistd.h>
#include<sys/wait.h>
#include <netinet/in.h>
#include <sys/socket.h>
#include <arpa/inet.h>
int main(int x, char* argv[]) {
char buf[100],data[100];
memset(buf,'\0',100);
struct sockaddr_in server, client;
int s_check, c_check;
s_check = socket(AF_INET, SOCK_STREAM, 0);
server.sin_family = AF_INET;
server.sin_addr.s_addr = INADDR_ANY;
server.sin_port = htons(atoi(argv[1]));
bind(s_check,(struct sockaddr *)&server, sizeof(server));
listen(s_check, 2);
int size = sizeof(client);
int pid=fork();
while(1)

{

c_check = accept(s_check, (struct sockaddr*)&client, &size);

memset(buf,'\0',100);

while(1) {

printf("\n entr data");

gets(data);

strcpy(buf,data);

send(c_check, buf, 100, 0);

if(strcmp(buf, "BYE")==0)

break;

printf("\nClient IP address is: %s\n", inet_ntoa(client.sin_addr));

printf("\nLocal port is: %d\n", (int) ntohs(client.sin_port));

recv(c_check, buf, 100, 0);

printf("\nRecieved data is : %s\n", buf);

if(strcmp(buf, "BYE")==0)

break;

}

close(c_check);

}

close(s_check);


return(0);

}
```