Program 3 Checksum program.

R Akash
1RV18CS125
NPS LAB Internal.

```
#include <stdio.h>
unsigned fields [10];
unsigned short checksum () {
        int i;
        int sum=0;
        printf("Enter IP header information in 16 bit words");
        for(i=0; i<9; i++) {
            printf("Field %d\n", i+1);
            scanf("%x", &fields [i]);
            sum = sum + (unsigned short) fields [i];
            while (sum >>16)
                sum = (sum & 0xFFFF) + (sum>>16);
        }
        sum = ~sum;
        return (unsigned short) sum;
}
int main () {
    unsigned short result1, result2;
    // Sender
    result1 = checksum ();
    printf("\n Computed checksum at sender %x\n", result1);
    //Receiver
    result2 = checksum ();
    printf("\n Computed checksum at receiver %x\n", result2);
```

```
if (result1 == result2)
    printf("No error");
else
    printf("Error in data received");
}
```

HAMMING CODE PROGRAM

```
#include<stdlib.h>
#include<stdio.h>
int main() {
    int a[4], b[4], r[3], s[3], i, q[3], c[7];
    printf("\n enter 4 bit data word:\n");
    for (i=3; i>=0; i--)
        scanf("%d", &a[i]);
    r[0] = (a[3] + a[1] + a[0]) %2;
    r[1] = (a[0] + a[2] + a[3]) %2;
    r[2] = (a[1] + a[2] + a[3]) %2;
    printf("\n The 7 bit hamming code is:\n");
    for(i=3; i>=0; i--)
        printf("%d\t", a[i]);
    for (i=2; i>=0; i--)
        printf("%d\t", r[i]);
```

```
printf("\n Enter the 7 bit received codeword: ");
for(i=7;i>0;i--)
    scanf("%d",&c[i]);
b[3]=c[7];b[2]=c[6];b[1]=c[5];b[0]=c[4];r[2]=c[3];r[1]=c[2];
r[0]=c[1];
s[0]=(b[0]+b[1]+b[3]+r[0])%2;
s[1]=(b[0]+b[2]+b[3]+r[1])%2;
s[2]=(b[1]+b[2]+b[3]+r[2])%2;
printf("Syndrome is :\n");
for(i=2;i>=0;i--)
    printf("%d",s[i]);
if((s[2]==0)&&(s[1]==0)&&(s[0]==0))
    printf("Received error free\n");
if((s[2]==1)&&(s[1]==1)&&(s[0]==1)){
    printf("\n Error in received codeword, position-7th bit from
        right\n");
    [redacted] c[7]^=c[7]; flag=true;
    [redacted]
}
if((s[2]==1)&&(s[1]==1)&&(s[0]==0)){
    printf("\n Error in received codeword, position-6th bit
        from right\n");
    c[6]^=c[6]; flag=true;
}
if((s[2]==1)&&(s[1]==0)&&(s[0]==1)){
```

```
printf("\n Error in received codeword, position - 5th bit from
        right\n");
c[5]^=c[5]; flag=true;
}
if((s[2]==1)&&(s[1]==0)&&(s[0]==0)){
    printf("\n Error in received codeword, position - 4th bit from
        right\n");
    c[4]^=c[4]; flag=true;
}
if((s[2]==0)&&(s[1]==1)&&(s[0]==1)){
    printf("\n Error in received codeword, position - 3rd bit from
        packet\n");
    c[3]^=c[3]; flag=true;
}
if((s[2]==0)&&(s[1]==1)&&(s[0]==0)){
    printf("\n Error in received codeword, position - 2nd bit from
        packet\n");
    c[2]^=c[2]; flag=true;
}
if((s[2]==0)&&(s[1]==0)&&(s[0]==1)){
    printf("\n Error in received codeword position - 1st bit
        from packet\n");
    c[1]^=c[1]; flag=true;
}
if(flag==true){
  printf("\n Error in code received, so correct codeword is\n");
```

```
for (i=7; i>0; i--)
    printf("%d\t", c[i]);
return (1);
}
```