```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

//?????? ???? ???? ??? BinTreeElementType.
typedef struct
{
    char data[20];
    char dummy;
} BinTreeElementType;

typedef struct BinTreeNode *BinTreePointer;
 struct BinTreeNode {
    BinTreeElementType Data;
    BinTreePointer LChild, RChild;
} ;

typedef enum {
    FALSE, TRUE
} boolean;

//??????????? ??? ?? HuffmanDecoding.c .
void BuildDecodingTree(BinTreePointer *Root,   FILE  *CodeFile);
void AddToTree(char ch[], char code[], int codelen, BinTreePointer *Root);
void Decode(BinTreePointer Root, FILE* MessageFile);

int main()
{
    //?????? ??????????.
    FILE *CodeFile,*MessageFile;
    BinTreePointer ARoot;

    //??????? ??? ???????.
    CodeFile = fopen("codesRW.txt","r");

    //?? ??????? ?????? ???? ?? ???????,???????? ???????????? ?????????.
    if(CodeFile == NULL)
        printf("Can't open codesRW.txt\n");
    else
    {
        //?????? ????? ??? ?????????? BuildDecodingTree ??? ??? ??????????
        //??? ???.
        BuildDecodingTree(&ARoot, CodeFile);

        //??????? ??? ??????? ?? ?? ?????????????? ??????.
        MessageFile = fopen("program.txt","r");

        //?? ??????? ???? ??? ???????,???????? ???????????? ?????????.
        if(MessageFile == NULL)
            printf("Can't oped program.txt\n");
        else
        //?????? ????? ?????????? Decode ??? ??????????????? ??? ?????????.
            Decode(ARoot,MessageFile);
    }

    //???????? ???????.
    fclose(CodeFile);
    fclose(MessageFile);

    system("PAUSE");
}


/*???? ?? ??????????? ???????????? ?? ???? ?? HuffmanDecoding.c,
????? ????? ?? ??????????? ??????? ???? ?? ?????????? ??? ???????? ????
```

```
???? ??? ???? ????????? ??? ???? ????.*/
void BuildDecodingTree(BinTreePointer *Root,   FILE  *CodeFile)
{
    //?????? ???? ???? ??? ???? ????????? ?? ???????? ??? ???????? ????.
    char ch[20],code[10],termch;
    int nscan;

    (*Root) = (BinTreePointer)malloc(sizeof (struct BinTreeNode));
    (*Root)->Data.dummy = '*';
    (*Root)->LChild=NULL;
    (*Root)->RChild=NULL;

    while(TRUE)
    {
        /*?????? ???? ?? ???????? ??? ???? ??? ??? ???????????? ???(? ?????
         ????????? ???? ??? ???? ?????) ??? ?????????? ???? ???????? ??????????.*/
        nscan = fscanf(CodeFile, "%20[^\n]%c\n%10[^\n]\n",ch,&termch,code);

        if(nscan == EOF ) break;
        if(nscan != 3 || termch != '\n')
        {
            printf("Improper file format.\n");
            break;
        }
        else
            AddToTree(ch,code,strlen(code),&(*Root));

    }

}
 //?????? ???? ???? ??? ???? ????????? ?? ??????? ??? ???????? ????.
void AddToTree(char ch[], char code[], int codelen, BinTreePointer *Root)
{
    int i = 0;
    BinTreePointer TempPtr,p = *Root;

    while(i < codelen)
        if(code[i] == '0')
        {
            if(p->LChild == NULL)
            {
                TempPtr = (BinTreePointer)malloc(sizeof (struct BinTreeNode));
                strcpy(TempPtr->Data.data,"*");
                TempPtr->Data.dummy = '*';
                TempPtr->LChild=NULL;
                TempPtr->RChild=NULL;
                p->LChild=TempPtr;
            }
            i++;
            p = p->LChild;
        }
        else
            if(code[i] == '1')
            {
                if(p->RChild == NULL)
                {
                    TempPtr = (BinTreePointer)malloc(sizeof (struct BinTreeNode));
                    strcpy(TempPtr->Data.data,"*");
                    TempPtr->Data.dummy = '*';
                    TempPtr->LChild=NULL;
                    TempPtr->RChild=NULL;
                    p->RChild = TempPtr;
                }
                i++;
                p = p->RChild;
            }
            //?????? ???? ?? ??????? ??? ???? ???? ??? ??????.
            strcpy(p->Data.data,ch);
}

void Decode(BinTreePointer Root, FILE* MessageFile)
{
    char bit;
    BinTreePointer p;

    while(TRUE)
    {
        p = Root;

        while(p->LChild != NULL || p->RChild != NULL)
        {
            bit = fgetc(MessageFile);

            if(bit != EOF)
            {
                printf("%c",bit);

                if(bit == '0')
                    p = p->LChild;
                else
                    if(bit == '1')
                        p=p->RChild;
                    else
                        if(bit != '\n')
                            printf("MH APODEKTO bit:%c \n",bit);

            }
            else break;
        }
        if(bit != EOF ) printf("----%s\n",p->Data.data);
        else break;
    }
}
```