#include <stdio.h>  
#include <string.h>  
#include <stdlib.h>  
  
//#define ARR\_len(x) sizeof(x) / sizeof(x)[0]  
  
#include <math.h>  
#include <ctype.h>  
#include <io.h>  
#include <time.h>  
  
static char DivideR[8][9] = {{""}};  
  
static int DivideR\_len = 0;  
  
int IntegerLen(const int x) {  
 int n = x;  
 int r = 0;  
 while (1) {  
 int b = n / 10;  
 r++;  
 n = b;  
 if (!b) break;  
 }  
 return r;  
}  
  
char \*NewFileName(char \*Dest, const char \*filePath) {  
 char \*r = Dest;  
 char newFN[strlen(filePath) + 5];  
 int x = 2;  
 while (1) {  
 char xS[IntegerLen(x) + 1];  
 itoa(x, xS, 10);  
 strcpy(newFN, filePath);  
 strcat(newFN, " (");  
 strcat(newFN, xS);  
 strcat(newFN, ")");  
 if (access(newFN, F\_OK) == EOF) break;  
 x++;  
 }  
 strcpy(r, newFN);  
 return Dest;  
}  
  
long long getFileSize(FILE \*fp) {  
 long long sz;  
 fseek(fp, 0L, SEEK\_END);  
 sz = (long long) ftell(fp);  
 if (sz == -1) {  
 sz = \_ftelli64(fp);  
 }  
 fseek(fp, 0L, SEEK\_SET);  
 return sz;  
}  
  
char \*NumStr\_lenTo(char \*Dest, const char \*source, const int len\_to) {  
 char \*r = Dest;  
 char s[len\_to + 1];  
 strcpy(s, "");  
 for (int i = 0; i < len\_to - strlen(source); ++i) {  
 strcat(s, "0");  
 }  
 strcat(s, source);  
 strcpy(r, s);  
 return r;  
}  
  
char \*substring(char \*Dest, const char \*source, const int beginIndex, const int endIndex) {  
 char \*r = Dest;  
 strncpy(r, source + beginIndex, (size\_t) (endIndex - beginIndex));  
 return Dest;  
}  
  
int String\_56\_DivideInto(const char \*source, const int per\_char\_split\_\_7\_or\_8) {  
 int source\_l = strlen(source);  
 if (source\_l != 56 && source\_l != 57) return 1;  
 DivideR\_len = 56 / per\_char\_split\_\_7\_or\_8;  
 for (int i = 0; i < DivideR\_len; ++i) {  
 substring(DivideR[i], source, per\_char\_split\_\_7\_or\_8 \* i, per\_char\_split\_\_7\_or\_8 \* (i + 1));  
 }  
 return 0;  
}  
  
/\*void printArr(const char \*a, const int length) {  
 int l = length;  
 printf("[");  
 for (int i = 0; i < l; ++i) {  
 printf("%i", (int) a[i]);  
 if (i != l - 1) {  
 printf(",");  
 }  
 }  
 printf("]\n");  
}\*/  
  
int BinToDec(const char \*NumStr) {  
 int r = 0;  
 int j = 0;  
 for (int i = strlen(NumStr) - 1; i >= 0; --i) {  
 r += (NumStr[i] == '0' ? 0 : 1) \* pow((double) 2, (double) j);  
 j++;  
 }  
 return r;  
}  
  
char \*ToUpperCase(char \*Dest, const char \*string) {  
 char \*p = Dest;  
 int len = strlen(string);  
 char r[len + 1];  
 int i = 0;  
 while (1) {  
 r[i] = (char) toupper((int) string[i]);  
 if (string[i] == '\0') break;  
 i++;  
 }  
 strcpy(p, r);  
 return Dest;  
}  
  
int main(/\*const \*/int argc, char \*argv[]) {  
// argc = 3;  
 if (argc < 3) {  
 printf("%s", "Base128\nCommand [-encode | -decode] [[filePath]] [[DestFilePath]] |\n"  
 "Command [-encode | -decode] [[filePath]]");  
 return 'P';  
 }  
 char dA[strlen(argv[1]) + 1];  
 ToUpperCase(dA, argv[1]);  
 int d = (!strcmp(dA, "-DECODE")) ? 1 : 0;  
 char FileName[strlen(argv[2]) + 1], DestFileName[strlen(argv[argc == 3 ? 2 : 3]) + 1 + ((argc == 3) ? 4 : 0)];  
 strcpy(FileName, argv[2]);  
 char zFN[strlen(argv[2]) + 1 + ((argc == 3) ? 4 : 0)];  
 if (argc == 3) {  
 NewFileName(zFN, argv[2]);  
 strcpy(DestFileName, zFN);  
 } else if (argc == 4) {  
 strcpy(DestFileName, argv[3]);  
 }  
 FILE \*fp = NULL;  
 if ((fp = fopen(FileName, "rb")) == NULL) {  
 return 1;  
 }  
 FILE \*fpO = NULL;  
 if ((fpO = fopen(DestFileName, "wb")) == NULL) {  
 return 1;  
 }  
 long long fileSize = getFileSize(fp);  
 if (!d) {  
 printf("%s", "Encoding...\n");  
 clock();  
 long long a = fileSize / 7LL;  
 int b = (int) (fileSize % 7LL);  
 char b1[8] = {0, 0, 0, 0, 0, 'z', 'h', 'c'};  
 b1[0] = (char) b;  
 fwrite(b1, 8, 1, fpO);  
 char r[7] = {""};  
 for (long long i = 0; i < a; ++i) {  
 fread(r, 7, 1, fp);  
 if (i % 374490LL == 0) {  
 printf("progress: %f%%\n", (float) 100 \* ((float) (7 \* (i + 1)) / (float) fileSize));  
 }  
 static char s0[57] = {""};  
 static char itoaR[9] = {""};  
 strcpy(s0, "");  
 for (int k = 0; k < 7; ++k) {  
 itoa(((int) r[k]) & 0xFF, itoaR, 2);  
 char len\_To\_8R[9] = {""};  
 NumStr\_lenTo(len\_To\_8R, itoaR, 8);  
 strcat(s0, len\_To\_8R);  
 }  
 String\_56\_DivideInto(s0, 7);  
 char wR[8] = {""};  
 for (int l = 0; l < DivideR\_len; ++l) {  
 wR[l] = (char) BinToDec(DivideR[l]);  
 }  
 fwrite(wR, 8, 1, fpO);  
 }  
 if (b != 0) {  
 memset(r, 0, 7);  
 fread(r, 7, 1, fp);  
 static char s0[57] = {""};  
 static char itoaR[9] = {""};  
 strcpy(s0, "");  
 for (int k = 0; k < 7; ++k) {  
 itoa(((int) r[k]) & 0xFF, itoaR, 2);  
 char len\_To\_8R[9] = {""};  
 NumStr\_lenTo(len\_To\_8R, itoaR, 8);  
 strcat(s0, len\_To\_8R);  
 }  
 s0[56] = 0;  
 String\_56\_DivideInto(s0, 7);  
 char wR[8] = {""};  
 for (int l = 0; l < DivideR\_len; ++l) {  
 wR[l] = (char) BinToDec(DivideR[l]);  
 }  
 fwrite(wR, 8, 1, fpO);  
 }  
 clock\_t eT = clock();  
 printf("progress: 100%%\n"  
 "Finished!\n"  
 "time-taken: %lf s\n"  
 "Speed is about %lf MB/s", (double) eT / 1000,  
 ((double) fileSize) / (1048576) / ((double) eT / (double) 1000));  
 } else {  
 printf("%s", "Decoding...\n");  
 clock();  
 fileSize -= 8LL;  
 long long a = fileSize / 8LL;  
 char rF[1] = {""};  
 char r[8] = {""};  
 fread(rF, 1, 1, fp);  
 fread(r, 7, 1, fp);  
 char FF = rF[0];  
 if (!(r[4] == 'z' && r[5] == 'h' && r[6] == 'c')) {//  
 FF = '\0';//  
 fseek(fp, 0L, SEEK\_SET);//  
 }//  
 strcat(r, "");  
 long long l\_i\_t = a - 1;  
 for (long long i = 0; i < l\_i\_t; ++i) {  
 fread(r, 8, 1, fp);  
 if (i % 327679LL == 0) {  
 printf("progress: %f%%\n", (float) 100 \* ((float) (7 \* (i + 1)) / (float) fileSize));  
 }  
 static char s0[57] = {""};  
 static char itoaR[8] = {""};  
 static char len\_To\_7R[9] = {""};  
 strcpy(s0, "");  
 for (int j = 0; j < 8; ++j) {  
 itoa((int) r[j] & 0xFF, itoaR, 2);  
 NumStr\_lenTo(len\_To\_7R, itoaR, 7);  
 strcat(s0, len\_To\_7R);  
 }  
 String\_56\_DivideInto(s0, 8);  
 char wR[7] = {""};  
 for (int k = 0; k < DivideR\_len; ++k) {  
 wR[k] = (char) BinToDec(DivideR[k]);  
 }  
 fwrite(wR, 7, 1, fpO);  
 }  
 fread(r, 8, 1, fp);  
 static char s0[57] = {""};  
 static char itoaR[8] = {""};  
 static char len\_To\_7R[9] = {""};  
 strcpy(s0, "");  
 for (int j = 0; j < 8; ++j) {  
 itoa((int) r[j] & 0xFF, itoaR, 2);  
 NumStr\_lenTo(len\_To\_7R, itoaR, 7);  
 strcat(s0, len\_To\_7R);  
 }  
 String\_56\_DivideInto(s0, 8);  
 char wR[7] = {""};  
 for (int k = 0; k < DivideR\_len; ++k) {  
 wR[k] = (char) BinToDec(DivideR[k]);  
 }  
 fwrite(wR, (size\_t) (FF == '\0' ? 7 : FF), 1, fpO);  
 clock\_t eT = clock();  
 printf("progress: 100%%\n"  
 "Finished!\n"  
 "time-taken: %lf s\n"  
 "Speed is about %lf MB/s", (double) eT / 1000,  
 ((double) fileSize) / (1048576) / ((double) eT / (double) 1000));  
 }  
 fclose(fp);  
 fclose(fpO);  
 if (argc == 3) {  
 remove(argv[2]);  
 rename(zFN, argv[2]);  
 }  
 return 0;  
}