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

// Check is program args contains string to encode
int isInputStringProvided(int argc, const char * argv[]) {
    if(argc < 2) {
        printf("no input sting to encode\n");
        return -1;
    }
    else if(argc > 2) {
        printf("please provide 1 input string to encode\n");
        return -1;
    }
    return 0;
}

// Check is string to encode is correct and adjust it for further processing
int isInputStringCorrect(const char * inputString, char * outputString) {
    //Convert input letters to low letters
    int i;
    for(i = 0; inputString[i]; i++) {
        outputString[i] = tolower(inputString[i]);
    }
    outputString[i] = '\0';
    //Check if there are only letters in string
    for(int i = 0; inputString[i]; i++) {
        if( (outputString[i] < 'a') || (outputString[i] > 'z')) {
            printf("please provide input string containing only letters\n");
            outputString = NULL;
            return -1;
        }
    }

    return 0;
}

// Perform encoding
void encode(const char * inputString, char * outputString) {

    char prevChar = 0;//a
    int counter = 0;//1
    char buffer[10];
    sprintf(buffer, "");

    for(int i = 0; inputString[i]; i++) {
        if(inputString[i] != prevChar) {
            //first sign
            if(prevChar == 0) {
                prevChar = inputString[i];
                counter = 1;
            }
            //new sign
            else {
                sprintf(buffer, "%i", counter);
                strcat(outputString, buffer);
                sprintf(buffer, "%c", prevChar);
                strcat(outputString, buffer);

                counter = 1;
                prevChar = inputString[i];
            }
        }
        //the same sign
        else if(inputString[i] == prevChar) {
            counter++;
        }
    }
    if((prevChar != 0) && (counter != 0)) {
        sprintf(buffer, "%i", counter);
        strcat(outputString, buffer);
        sprintf(buffer, "%c", prevChar);
        strcat(outputString, buffer);
```

```
74          strcat(outputString, "\0");
75      }
76  }
77  // Perform decoding
78  void decode(const char * inputString, char * outputString) {
79
80      int value = 0;
81      char currChar = 0;
82
83      printf("decode: input string: %s\n", inputString);
84
85      for(int i = 0; inputString[i]; i = i+2) {
86          if(inputString[i+1] == '\0')
87              return;
88          //read number
89          sscanf(&inputString[i], "%*[^0123456789]%d", &value);
90          printf("  value[%i]:%i\n", i, value);
91          //read sign
92          sscanf(&inputString[i+1], "%*[^abcdefghijklmnopqrstuvwqyz]%c", &currChar);
93          printf("  currChar[%i]:%c\n", i, currChar);
94
95          for(int j = 0; j < value; j++){
96              strncat(outputString, &currChar, 1);
97          }
98      }
99  }
100
101 int main(int argc, const char * argv[]) {
102
103     //Check input
104     if(isInputStringProvided(argc, argv) == -1) {
105         return -1;
106     }
107
108     //Create buffers
109     char inStr[strlen(argv[1])];
110     char outStr[strlen(argv[1])];
111     char outStr2[strlen(argv[1])];
112     sprintf(outStr2, "%c", '\0');
113
114     //Preprocess input
115     if(isInputStringCorrect(argv[1], inStr) == -1) {
116             printf("problems with input string");
117             return -1;
118     }
119
120     //Encode
121     if(inStr) {
122         printf("input string to encode is %s\n", inStr);
123         printf("Lets encode: \n");
124     }
125     encode(inStr, outStr);
126     printf("Encoded input is: %s\n", outStr);
127
128     //Decode back
129     if(outStr) {
130         printf("Lets decode back: \n");
131     }
132     decode(outStr, outStr2);
133     printf("Input decoded back is: %s\n", outStr2);
134
135     return 0;
136 }
```