**C Programming – Final Project**

By: Rishi Agarwal

**Prologue:** Encoding and decoding play a vital role in modern communication and data storage. With the ever-increasing amount of data being transmitted and stored, the need for efficient and secure methods of encoding and decoding is more pressing than ever. Encoding and decoding in is a method of representing data in a different format to efficiently transfer information through a network or the web. The encoder converts data into a web representation. Once received, the decoder converts the web representation data into its original format

**Aim:** The aim of this project is to provide a simple and flexible solution for encoding and decoding any inputted string or contents of a file. This can be used for various purposes such as securing sensitive information, or transforming data into a format that is easier to transmit or store.

**Idea:** The project uses a basic three-level operation to encode and decode the input, providing a simple and fast method of transforming data. The program is written in C and is designed to be easy to use and understand, making it suitable for a wide range of applications. The project first converts the given contents to their ASCII value and then this value is converted to a random base number system with all the numerical digits converted to symbols.

**Program:**

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <time.h>

#include <math.h>

#include <conio.h>

int generate\_random\_base() {

    srand(time(0));

    int base = (rand() % (10 - 2 + 1)) + 2;

    return base;

}

int BaseToAscii (long int number, int base) {

    long int temp = number;

    int ascii\_value = 0;

    int temp\_count = 0;

    while (temp!=0) {

        int last\_digit = temp%10 ;

        ascii\_value += (last\_digit \* pow(base,temp\_count));

        ++temp\_count;

        temp = temp/10;

    }

    return ascii\_value;

}

void delay(int number\_of\_seconds) {

    int milli\_seconds = 1000 \* number\_of\_seconds;

    clock\_t start\_time = clock();

    while (clock() < start\_time + milli\_seconds)

        ;

}

int main() {

    char test\_string [256];

    printf("\nEnter the string to be encrypted: ");

    fgets(test\_string, 255, stdin);

    int n = strlen(test\_string);

    int encryption\_base = generate\_random\_base() ;

    printf("\nYour Key is %d. Please remember it otherwise you can't decrypt your message.\n", encryption\_base);

    printf("\nEncrypting..\n");

    delay(3);

    int final\_length = n\*16;

    int encrypted\_msg [final\_length] ;

    int pos=0;

    for (int i=0; i<n; ++i) {

        char c = test\_string[i];

        int c\_ascii = (int)c;

        int reverse\_converted [16];

        int temp = c\_ascii;

        int n2=0;

        int subscript = 0;

        while (temp>0) {

            reverse\_converted[subscript] = (temp%encryption\_base);

            temp = temp/encryption\_base;

            n2++;

            ++subscript;

        }

        int final\_converted [n2];

        int k=0;

        for (int j = n2 - 1; j >= 0; --j) {

            final\_converted[k] = reverse\_converted[j];

            ++k;

        }

        for (int l=0; l<n2; ++l) {

            encrypted\_msg[pos+l]=final\_converted[l];

        }

        pos+=n2;

        encrypted\_msg[pos] = 9999;

        pos+=1;

    }

    char super\_encrypted\_msg [pos];

    for (int g=0; g<pos; ++g) {

        if (encrypted\_msg[g] == 0)

            super\_encrypted\_msg[g] = ')' ;

        else if (encrypted\_msg[g] == 1)

            super\_encrypted\_msg[g] = '!' ;

        else if (encrypted\_msg[g] == 2)

            super\_encrypted\_msg[g] = '@' ;

        else if (encrypted\_msg[g] == 3)

            super\_encrypted\_msg[g] = '#' ;

        else if (encrypted\_msg[g] == 4)

            super\_encrypted\_msg[g] = '$' ;

        else if (encrypted\_msg[g] == 5)

            super\_encrypted\_msg[g] = '%' ;

        else if (encrypted\_msg[g] == 6)

            super\_encrypted\_msg[g] = '^' ;

        else if (encrypted\_msg[g] == 7)

            super\_encrypted\_msg[g] = '&' ;

        else if (encrypted\_msg[g] == 8)

            super\_encrypted\_msg[g] = '\*' ;

        else if (encrypted\_msg[g] == 9)

            super\_encrypted\_msg[g] = '(' ;

        else if (encrypted\_msg[g] == 9999)

            super\_encrypted\_msg[g] = '~' ;

    }

    printf("\nYour encrypted message is:\n");

    for (int m=0; m<pos; ++m)  {

        printf("%c ",super\_encrypted\_msg[m]);

    }

    printf("\n\nPress Enter to continue with decryption ");

    char temp\_choice;

    scanf("%c",&temp\_choice);

    int key ;

    printf("\n\nEnter your Decryption Key: ");

    scanf("%d",&key);

    if (key !=  encryption\_base) {

        printf("\nYou entered the wrong Key. You have only one chance left. Enter Again: ");

        scanf("%d",&key);

        if (key !=  encryption\_base) {

            printf("\nSorry you again entered the wrong key. You can't decrypt your message.");

            exit(1);

        }

    }

    // DECRYPTION :

    char temp\_encrypted [final\_length];

    for (int p=0; p<pos; ++p) {

        temp\_encrypted[p] = super\_encrypted\_msg[p];

    }

    printf("\n\n\nDecrypting...");

    delay(3);

    int first\_decrypted [pos];

    for (int q=0; q<pos; ++q)  {

        if (temp\_encrypted[q] == ')')

            first\_decrypted[q] = 0;

        else if (temp\_encrypted[q] == '!')

            first\_decrypted[q] = 1;

        else if (temp\_encrypted[q] == '@')

            first\_decrypted[q] = 2;

        else if (temp\_encrypted[q] == '#')

            first\_decrypted[q] = 3;

        else if (temp\_encrypted[q] == '$')

            first\_decrypted[q] = 4;

        else if (temp\_encrypted[q] == '%')

            first\_decrypted[q] = 5;

        else if (temp\_encrypted[q] == '^')

            first\_decrypted[q] = 6;

        else if (temp\_encrypted[q] == '&')

            first\_decrypted[q] = 7;

        else if (temp\_encrypted[q] == '\*')

            first\_decrypted[q] = 8;

        else if (temp\_encrypted[q] == '(')

            first\_decrypted[q] = 9;

        else if (temp\_encrypted[q] == '~')

            first\_decrypted[q] = 9999;

    }

    printf("\n\n");

    for (int r=0; r<pos; ++r) {

        printf("%d ",first\_decrypted[r]);

    }

    delay(1);

    int second\_decrypted [n];

    int second\_decrypted\_index = 0;

    int previous\_9999\_index = -1;

    for (int s=0; s<pos; ++s) {

        if (first\_decrypted[s] == 9999) {

            int current\_9999\_index = s;

            int len\_of\_s;

            if (previous\_9999\_index != -1) {

                len\_of\_s = current\_9999\_index - previous\_9999\_index - 1;

            } else {

                len\_of\_s = current\_9999\_index;

            }

            char temp\_str [len\_of\_s];

            int temp\_i=0;

            for (int t=previous\_9999\_index+1; t<current\_9999\_index; ++t) {

                char temp\_char = first\_decrypted[t]+'0';

                temp\_str[temp\_i] = temp\_char;

                ++temp\_i;

            }

            int base\_int\_value = atoi(temp\_str);

            second\_decrypted[second\_decrypted\_index] = base\_int\_value;

            ++second\_decrypted\_index;

            previous\_9999\_index = current\_9999\_index;

        }

    }

    printf("\n");

    for (int u=0; u<n; ++u) {

        printf("%d ",second\_decrypted[u]);

    }

    delay(1);

    int third\_decrypted [n];

    for (int v=0; v<n; ++v) {

        third\_decrypted[v] = BaseToAscii(second\_decrypted[v],encryption\_base);

    }

    printf("\n");

    for (int w=0; w<n; ++w) {

        printf("%d ",third\_decrypted[w]);

    }

    delay(1);

    char final\_decrypted [n-1];

    for (int z=0; z<n-1; ++z) {

        char temp\_c = third\_decrypted[z];

        final\_decrypted[z] = temp\_c;

    }

    printf("\n\nYour DECRYPTED message is : ");

    for (int y=0; y<n-1; ++y) {

        printf("%c",final\_decrypted[y]);

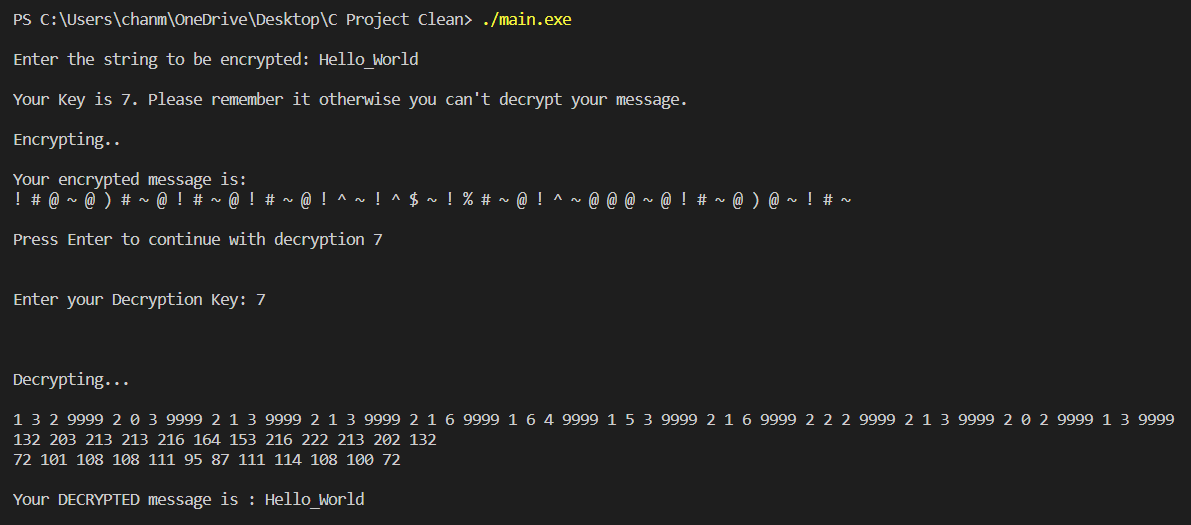
    }

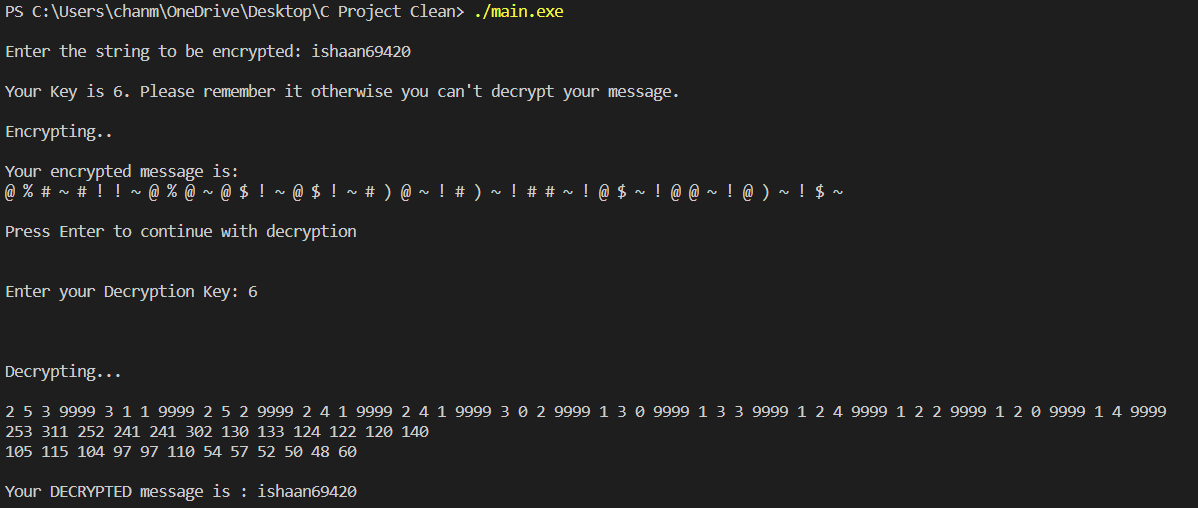
    printf("\n\n");

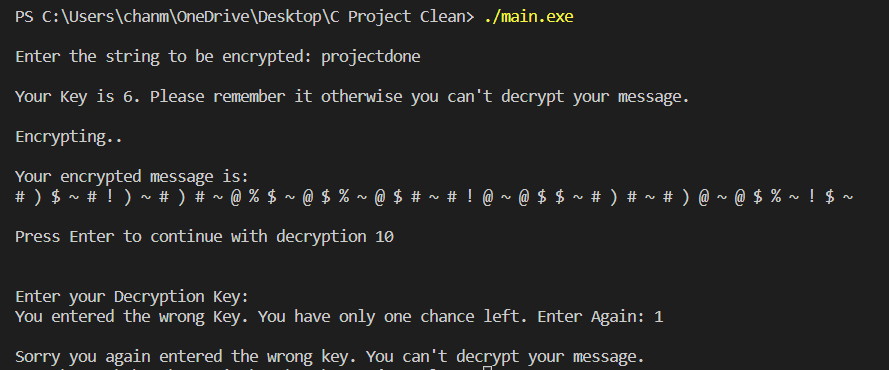
    return 0;

}

**Outputs:**

****

****

****

**Conclusion:** In conclusion, this project provides a useful tool for encoding and decoding data in today's fast-paced digital world, offering a simple and secure solution for protecting sensitive information and transforming data for more efficient storage and transmission.

***Thanks***

Team Members: **Chanmeet Singh** – 08 – K1 – CSE

**Ishaan Minocha** – 31 – K1 – CSE

**Haaris Huda** – 13 – K1 – CSE

**Rishi Agarwal** – 48 – K2 – CSE

--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------