PROJECT REPORT

On

File Encryption with C++

(CSE III Semester Mini project)

2020-2021

Submitted to:

Miss. Manika Manwal

(CC-CSE-E-III-Sem)

Guided by:

Mr.

(Resource Person)

Submitted by:

Mr. Mohit Sharma

Roll. No.: 1918483

CSE-E-III-Sem

Session: 2020-2021

DEPARTMENT OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

GRAPHIC ERA HILL UNVERSITY, DEHRADUN

CERTIFICATE

Certified that Mr. Mohit Sharma (Roll No.- 1918483) has developed mini project on “File Encryption with C++” for the CS III Semester Mini Project Lab (PCS-504) in Graphic Era Hill University, Dehradun. The project carried out by Students is their own work as best of my knowledge.

Date: 1-december-2020

(Miss. Manika Manwal)	(Dr. Ashook Sahoo)
Project Co-ordinator	**Project Guide**
CC-CSE-E-III-Sem	Resource Person
(CSE Department)	(CSE Department)
GEHU Dehradun	GEHU Dehradun

ACKNOWLEDGMENT

would like to express our gratitude to, the most Beneficent and the most Merciful, for completion of project.

We wish to thank our parents for their continuing support and encouragement. We also wish to thank them for providing us with the opportunity to reach this far in our studies.

We would like to thank particularly our project Co-ordinator Miss. Manika Manwal Dubey and our Project Guide Dr. Ashook Sahoo for his patience, support and encouragement throughout the completion of this project and having faith in us.

We also acknowledge to teachers who help us in developing the project.

At last but not the least We greatly indebted to all other persons who directly or indirectly helped us during this work.

Mr. Mohit Sharma

Roll No.- 1918483

CSE-E-III-Sem

Session: 2020-2021

GEHU, Dehradun We

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ALGORITHMS

CHAPTER 1

INTRODUCTION

1.1 ABOUT PROJECT

This Project is based on File Encryption and decryption process. In this project I've created a program that takes a file as command line input. Input is the name of any type of file that we want to encrypt or decrypt.

The main feature of my project are:

1. Protection Feature with Secret key

   When a user selects an option to encrypt a file in that case, he has to provide a secret key that will get stored in encrypted file at random position. This key value acts as a password. In case if user wants to decrypt the same file, then he has to provide the same key value or password to continue the file decryption process. In case password does not matches then the program exits with an error message.

2. While Encrypting and Decrypting this program also shows progress how much amount of data has been encrypted.

3.It is capable of encrypting as well as decrypting any type of file. For example, if file is abc.txt/dox/pdf then encrypted file will be abc.txt. ecrypt.

4.Maximum Size limit

   The maximum file size that this program support is 1 Gb.

This is a C++ based project and this project report contains in depth demonstration of Data Encryption and Decryption system along with its working architecture which contains brief introduction, features, outputs, source code, Algorithms and at last references.

1.2 WHAT IS ENCRYPTION AND DECRYPTION

Encryption is the process of taking plain text, like a text message or email, and scrambling It into an unreadable format – called “cipher text”.

When the intended recipient accesses the message, the information is translated back to its original form. This is called decryption.
To unlock the message, both the sender and the c=recipient have to use “secret” key – a collection of algorithm that scramble and unscramble data back to a readable format.

CHAPTER 2
PROJECT

2.1 REQUIREMENT ANALYSIS

Name: Simple Password Encryption in C++

Features: Uses simple algorithm to encrypt the file

Explanation: This Project use the concept of secret key which is to be given by user during encryption process. After getting the key, it is decrypted with the simple algorithm so that no other person than the authorized one can be able to view the content of file.

2.2 SOFTWARE SPECIFICATION

2.2.1 Tool Required and Requirement:

- Linux Environment
- CodeBlocks IDE
- Terminal
- Ram Minimum 512 MB

2.3 DATA FLOW DIAGRAM

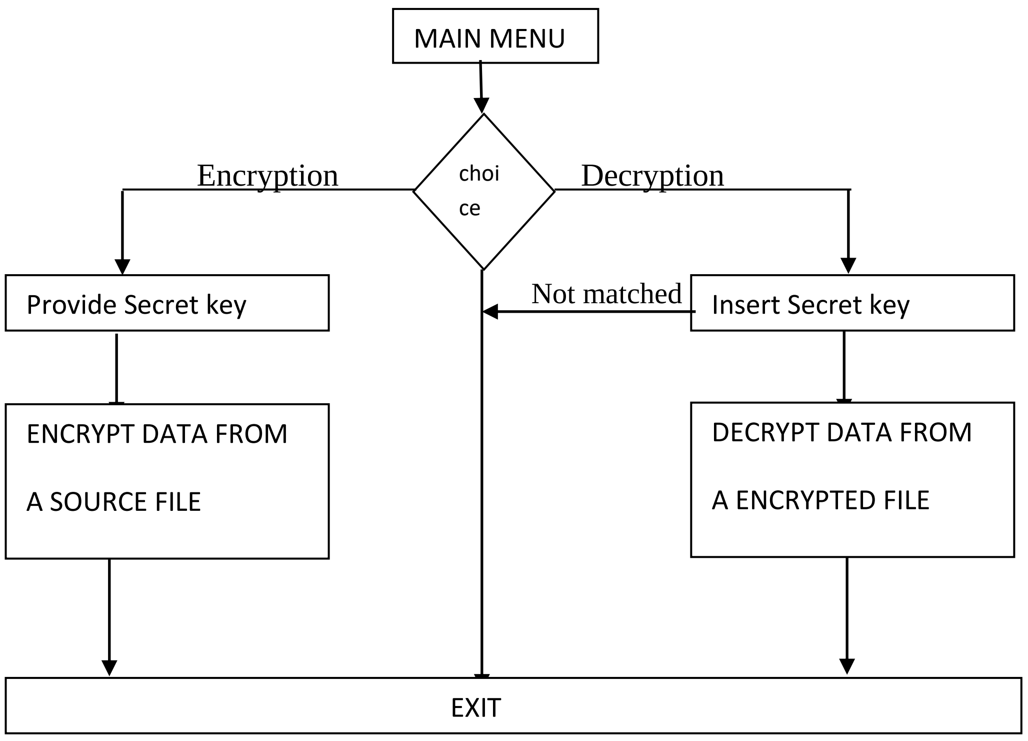
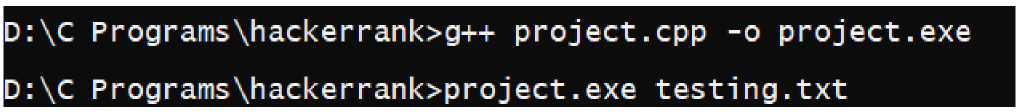


Figure 2.3.1 Flowchart of Project

2.4 MODULE

2.4.1 Entry of File.

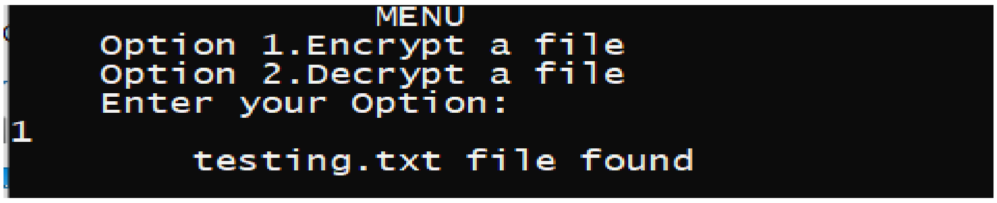
In the program file name with path is given as an command line input. If file is not found at the path entered by the user then the program displays an error message on console and exits with error otherwise executes further



```
D:\C Programs\hackerrank>g++ project.cpp -o project.exe

D:\C Programs\hackerrank>project.exe testing.txt
```

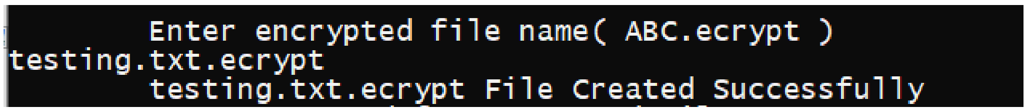
2.4.2 Selection of Option

After inputting the source file path, we have two options as shown in the figure. Choosing 1 option will led to encryption and second option will led to decryption. To select an option, the user will have to enter his choice. The feel of main menu is kept simpler so that anyone who doesn’t prior knowledge about encryption and decryption can also be able to use this easily.

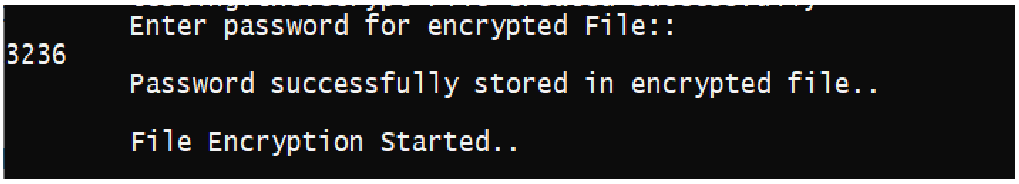

```
              MENU
    Option 1.Encrypt a file
    Option 2.Decrypt a file
    Enter your Option:
1
        testing.txt file found
```

2.4.3 Creating new file

If source file have been found successfully, then the program asks the user to enter the name of the new file in which encrypted data will gets stored. After processing it creates a new file named Abc.txt.ecrypt or Abc.txt.decrypt in the same directory which will contain the encrypted data. This Step is also similar in decryption process as well.


```
        Enter encrypted file name( ABC.ecrypt )
testing.txt.ecrypt
        testing.txt.ecrypt File Created Successfully
```

2.4.4 Providing a key/password

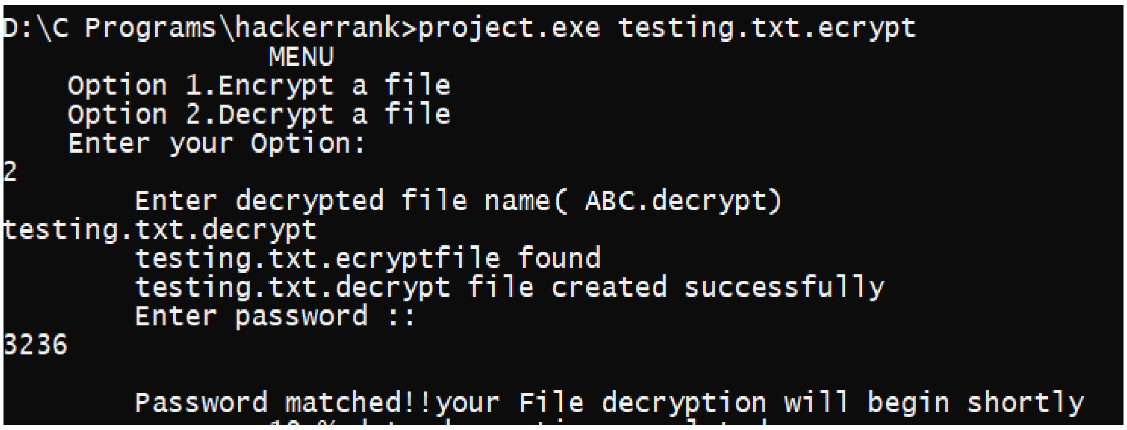

```
        Enter password for encrypted File::
3236
        Password successfully stored in encrypted file..

        File Encryption Started..
```

In next step user have to enter a key value, this key value can be anything. Let us take an example, suppose user have entered 326 then this 326 will added to each character of file. The key value and each character gets stored in the new encrypted file.

2.4.5 Decrypting the file

To decrypt the same file, it is necessary to remember the key value, if user have forgotten it then there is no way to access the content of encrypted file.

To decrypt the same file user, have to provide the file path again as a command line input and have to select an option of decrypt. Then the program asks for key, if the key entered by user is correct then a new file will get created with decrypt extension.



```
D:\C Programs\hackerrank>project.exe testing.txt.ecrypt
                MENU
    Option 1.Encrypt a file
    Option 2.Decrypt a file
    Enter your Option:
2
        Enter decrypted file name( ABC.decrypt)
testing.txt.decrypt
        testing.txt.ecryptfile found
        testing.txt.decrypt file created successfully
        Enter password ::
3236

        Password matched!!your File decryption will begin shortly
```

2.4 APPLICATION

- Encryption is the process of taking plain text, like a text message or email, and scrambling it into an unreadable format — called “cipher text.” This helps protect the confidentiality of digital data either stored on computer systems or transmitted through a network like the internet.

2.5 ALGORITHMS

2.6.1 Encryption process algorithm

```
1.  Start
2.  Input File name by user
3.  argv[1]=Abc.txt
4.  Open argv[1] file in read mode
5.  If (file not found)
    Goto step  16
6.  Fp1=starting location of the source file
7.  Create new file name abc.txt.ecrypt in write mode
8.  If (not created)
        a.  Goto step 16
9.  Fp2=starting location of encrypted file
10. Input a key by user
11. Store this key in file pointed by fp2
        fputc(key,fp2)
12. Repeat for i=0 to end of file
        ch=get integer from File pointed by fp1
        if(ch==EOF)
        goto to step 16
        otherwise
                ch=+key
                write ch into encrypted file pointed by fp2
                fp2=fp2+fp2
14.close source file
15.close encrypted file
16.stop
```

2.6.2 Decryption process algorithm

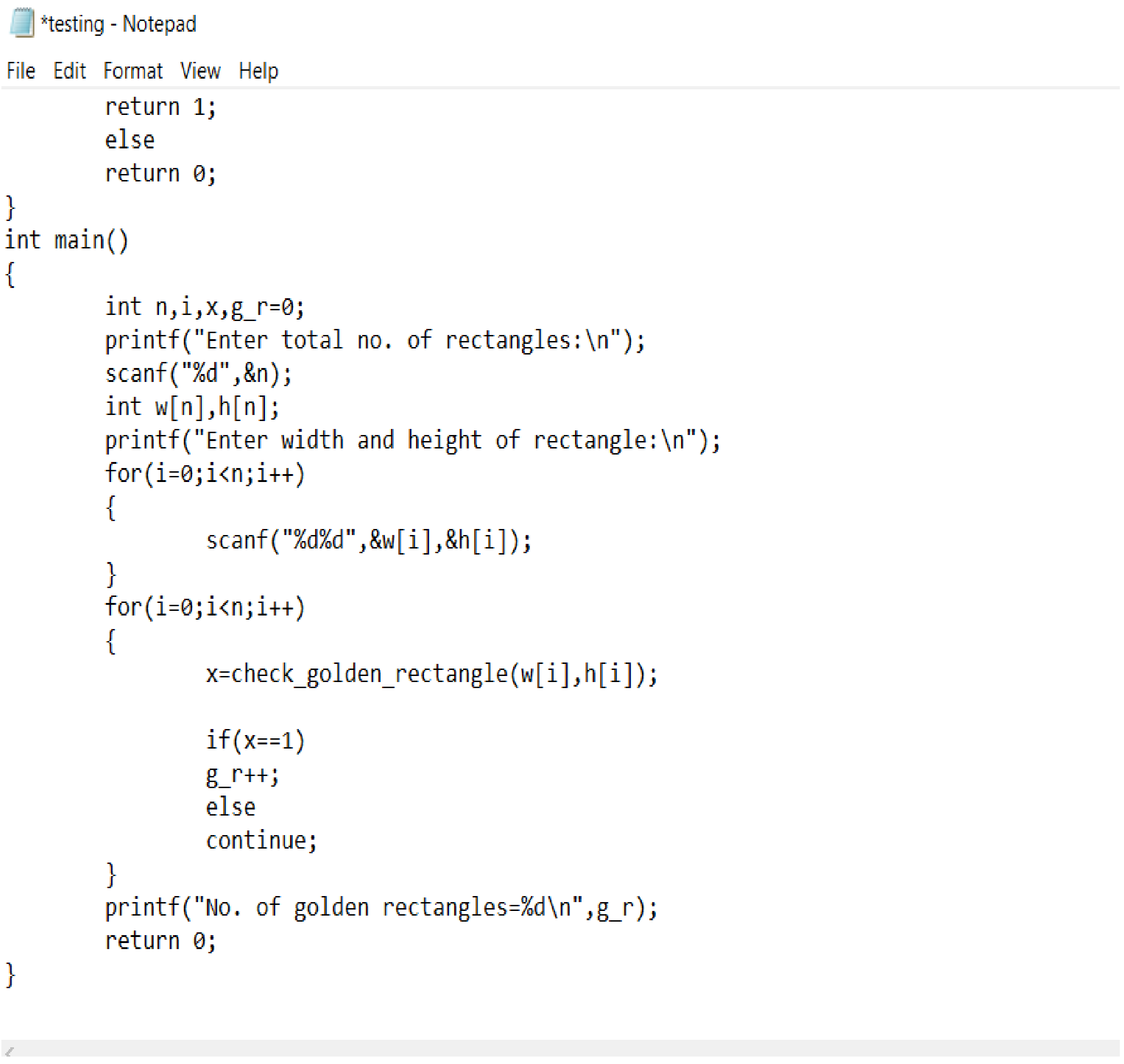
```
1.Start
2. Input file name by user
3.argv[1]=abc.txt.ecrypt
4.open argv[1] in read mode
```

```
5.if file not found
          Goto to step 16
6.fp1=starting location of source file argv[1]
7.create a new file as abc.txt.decrypt in write mode
8.if file not found
          Goto to step 16
9.fp2=starting location of decrypted file created
10. read key from fp1 file as key1
11.input key from user as key2
12.if key1!=key2
          Goto to step 16
13.Repeat for i=0 till EOF reached
          ch=get integer from File pointed by fp1
          if(ch==EOF)
                    goto to step 16
          otherwise
                    ch=-key
                    write ch into encrypted file pointed by fp2
                    fp2=fp2+fp2
14.close fp1
15.close fp2
16.Stop
```

CHAPTER 3

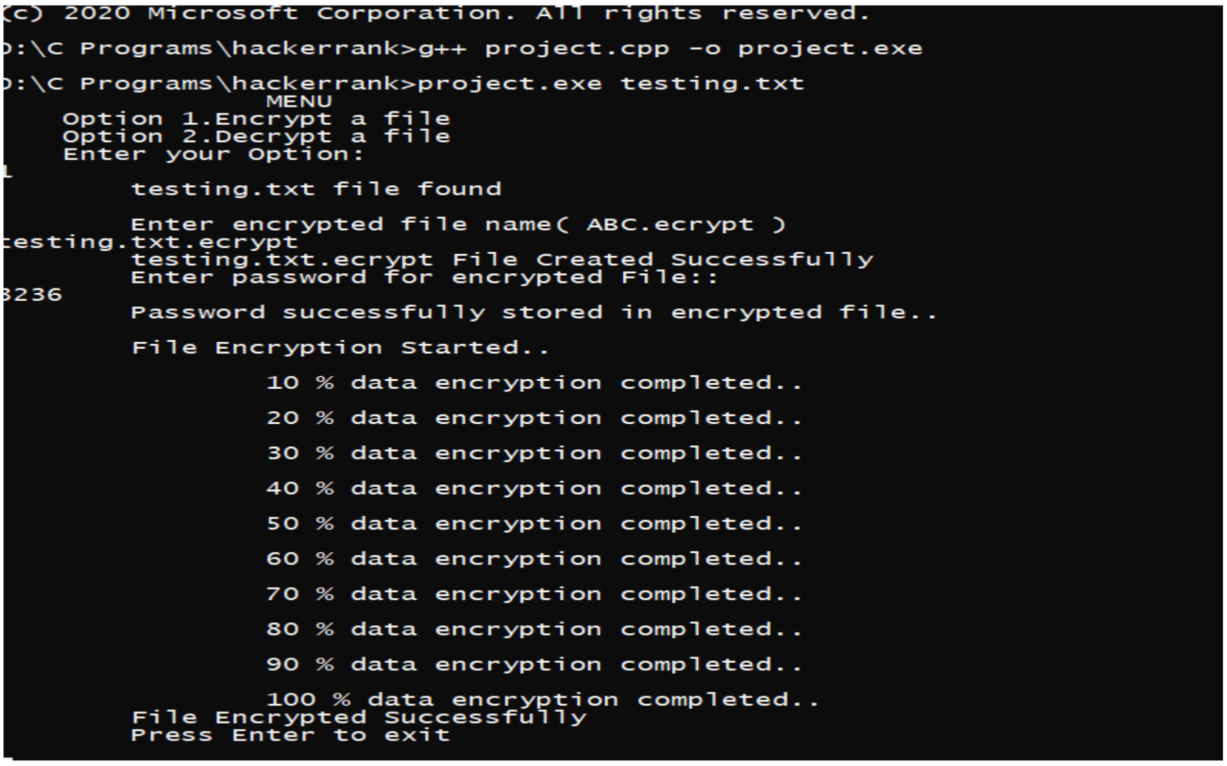
SNAPSHOT OF PROJECT

3.1 SOURCE FILE CONTENT



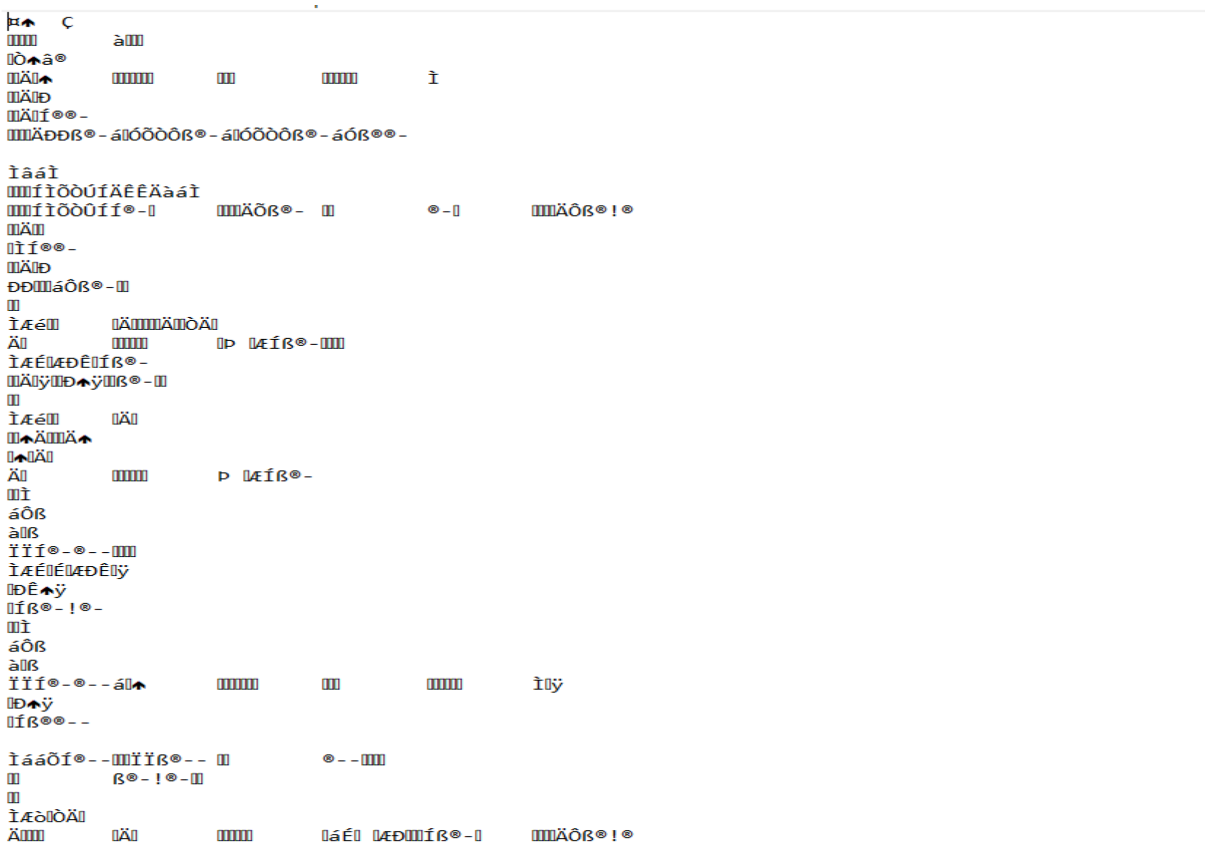
```
        return 1;
        else
        return 0;
}
int main()
{
        int n,i,x,g_r=0;
        printf("Enter total no. of rectangles:\n");
        scanf("%d",&n);
        int w[n],h[n];
        printf("Enter width and height of rectangle:\n");
        for(i=0;i<n;i++)
        {
                scanf("%d%d",&w[i],&h[i]);
        }
        for(i=0;i<n;i++)
        {
                x=check_golden_rectangle(w[i],h[i]);

                if(x==1)
                g_r++;
                else
                continue;
        }
        printf("No. of golden rectangles=%d\n",g_r);
        return 0;
}
```

3.2 ENCRYPTION PROCESS



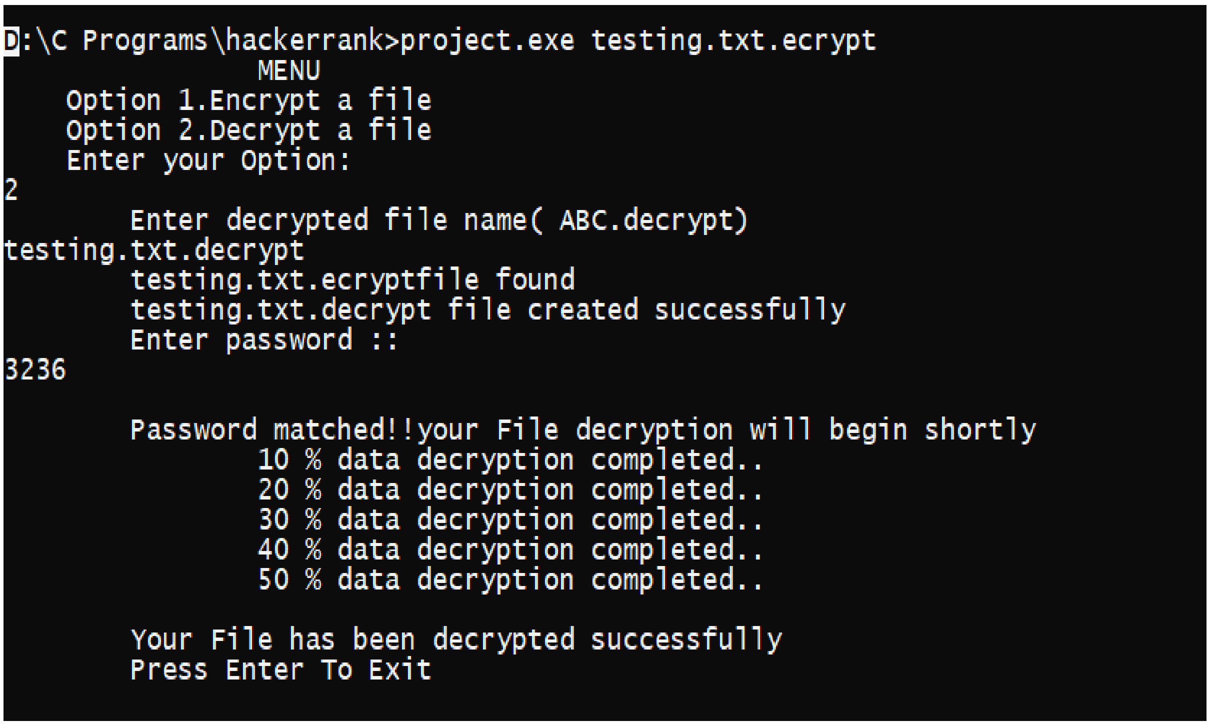
```
(c) 2020 Microsoft Corporation. All rights reserved.

D:\C Programs\hackerrank>g++ project.cpp -o project.exe

D:\C Programs\hackerrank>project.exe testing.txt
                    MENU
    Option 1.Encrypt a file
    Option 2.Decrypt a file
    Enter your Option:
1
        testing.txt file found

        Enter encrypted file name( ABC.ecrypt )
testing.txt.ecrypt
        testing.txt.ecrypt File Created Successfully
        Enter password for encrypted File::
3236
        Password successfully stored in encrypted file..

        File Encryption Started..

                10 % data encryption completed..

                20 % data encryption completed..

                30 % data encryption completed..

                40 % data encryption completed..

                50 % data encryption completed..

                60 % data encryption completed..

                70 % data encryption completed..

                80 % data encryption completed..

                90 % data encryption completed..

                100 % data encryption completed..
        File Encrypted Successfully
        Press Enter to exit
```

3.2 .1 ENCRYPTED FILE



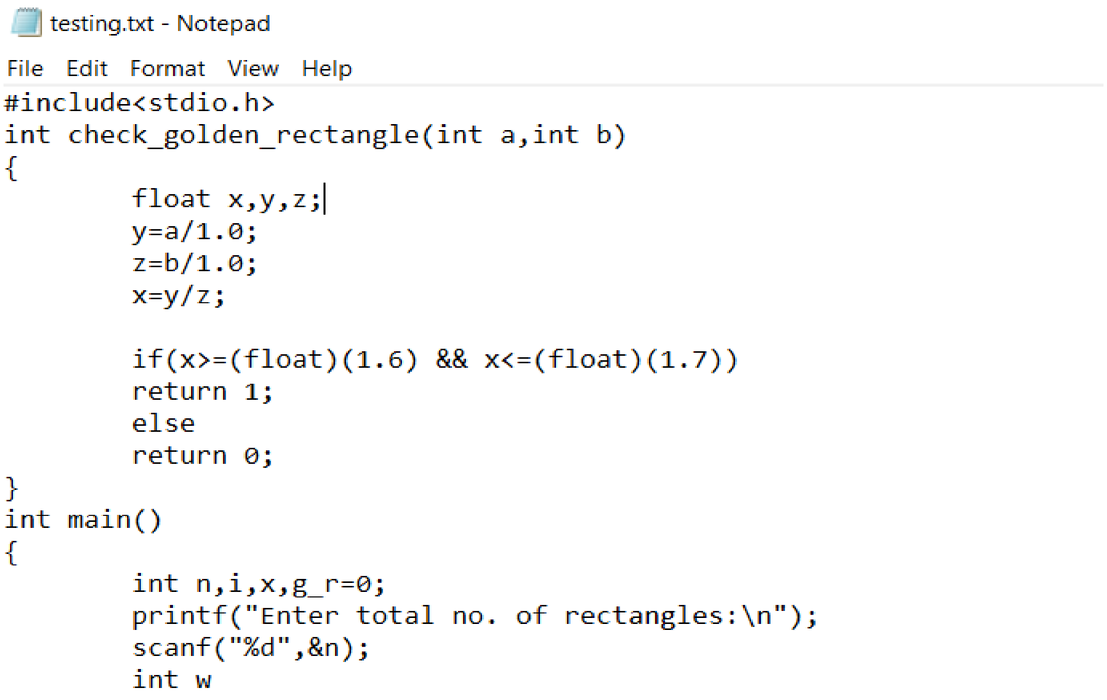
```
Ì¤▲   Ç
□□□□        à□□
□Ò▲â®
□□Ä□▲       □□□□□□□    □□□      □□□□□□     Ì
□□Ä□Ð
□□Ä□Í®®-
□□□ÄÐÐß®-á□ÓÕÒÒÔß®-á□ÓÕÒÒÔß®-áÓß®®-

ÌâáÌ
□□□ÍÌÕÒÚÍÄÊÊÄàáÌ
□□□ÍÌÕÒÙÍÍ®-□      □□□ÄÕß®-  □□       ®-□      □□□ÄÕß®!®
□□Ä□
□ÌÍ®®-
□□Ä□Ð
ÐÐ□□□áÔß®-□□
□□
ÌÆé□□      □Ä□□□□Ä□□ÒÄ□
Ä□         □□□□□□     □Þ □ÆÍß®-□□□□
ÌÆÉ□ÆÐÊ□Íß®-
□□Ä□ÿ□□Ð▲ÿ□□ß®-□□
□□
ÌÆé□□      □Ä□
□□▲Ä□□□Ä▲
□▲□Ä□
Ä□         □□□□□□     Þ □ÆÍß®-
□□Ì
áÕß
à□ß
ÏÏÍ®-®--□□□□
ÌÆÉ□É□ÆÐÊ□ÿ
□ÐÊ▲ÿ
□Íß®-!®-
□□Ì
áÕß
à□ß
ÏÏÍ®-®--á□▲      □□□□□□□    □□□      □□□□□□     Ì□ÿ
□Ð▲ÿ
□Íß®®--

ÌááÕÍ®--□□□ÏÏß®--  □□       ®--□□□□
□□         ß®-!®-□□
□□
ÌÆò□ÒÄ□
Ä□□□□      □Ä□        □□□□□□     □áÉ□ □ÆÐ□□□Íß®-□     □□□ÄÕß®!®
```

3.3 DECRYPTION PROCESS



```
D:\C Programs\hackerrank>project.exe testing.txt.ecrypt
                MENU
    Option 1.Encrypt a file
    Option 2.Decrypt a file
    Enter your Option:
2
        Enter decrypted file name( ABC.decrypt)
testing.txt.decrypt
        testing.txt.ecryptfile found
        testing.txt.decrypt file created successfully
        Enter password ::
3236

        Password matched!!your File decryption will begin shortly
                10 % data decryption completed..
                20 % data decryption completed..
                30 % data decryption completed..
                40 % data decryption completed..
                50 % data decryption completed..

        Your File has been decrypted successfully
        Press Enter To Exit
```

3.3.1 DECRYPTED FILE



```
#include<stdio.h>
int check_golden_rectangle(int a,int b)
{
        float x,y,z;
        y=a/1.0;
        z=b/1.0;
        x=y/z;

        if(x>=(float)(1.6) && x<=(float)(1.7))
        return 1;
        else
        return 0;
}
int main()
{
        int n,i,x,g_r=0;
        printf("Enter total no. of rectangles:\n");
        scanf("%d",&n);
        int w
```

CHAPTER 4

CONCLUSION

4.1 CONCLUSION

It can be concluded that this project works efficiently for offline purpose at any place provided the requirements for running environment remains same. Users need not to worry about how the algorithm actually works inside the instruction set performed by the compiler.

4.2 FUTURE SCOPE

1. In Future, Data Encryption and Decryption system can be updated further which can provide more secured encryption algorithms.
2. This project can be made more attractive and easy to use by implementing GUI(Graphical User Interface) with the help of many coding platforms like Netbeans etc.

APPENDIX

Code

```
#include<iostream>

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

#include<ctype.h>

#include<math.h>


using namespace std;


int file_size(FILE**fp1)

{

        fseek(*fp1,0L,2);          //point file pointer to end of source file

        int s=ftell(*fp1);

        //storing file size returned by ftell function

        if(s>(pow(2,30)))                  //checking file size

        {

        cout<<'\t'<<"Maximum size of file should not be greater than 1Gb"<<endl;
        cout<<endl<<'\t'<<"Press Enter to exit"<<endl;

        getch();

        exit(1);                           //end of program

        }

        rewind(*fp1);                             //point file pointer again at starting

        return s;
```

```
}
void encrypt_file(FILE **fp1,FILE **fp2,int key,int max_size)
{
        char ch;
        int i=10,curr_position,percent;
        //writing key value into new encrypted file//
        putw(key,*fp2);
        cout<<'\t'<<"Password successfully stored in encrypted file.."<<endl;
        cout<<endl<<'\t'<<"File Encryption Started.."<<endl;
        while(1)
        {
                ch=getc(*fp1);      //reading characters one by one
                if(ch==EOF)
                {
                cout<<'\t'<<"File Encrypted Successfully"<<endl;
                return;             //exit if end of file reached
                }//end of if block
                else
                {
                ch+=key;      //add key to  character and write it to file
                fputc(ch,*fp2);
                curr_position=ftell(*fp1);
                percent=(i*max_size)/100;
                if(curr_position==percent)
                {
                cout<<endl<<'\t'<<'\t'<<i<<" % data encryption completed.."<<endl;
                i+=10;
                }
```

```
                    }//end of else block

}//end of while loop

}//end of encryption function


void decrypt_file(FILE **fp1,FILE **fp2,int max_size)

{

          int key=getw(*fp1);

           //reading key stored in encrypted file

          char ch;

          int k,i=10,curr_position,percent;

          cout<<'\t'<<"Enter password :: "<<endl;

          cin>>k;

          if(key==k

)

          {

          cout<<endl<<'\t'<<"Password matched!!your File decryption will begin
          shortly"<<endl;

          while(1)

          {

                    ch=fgetc(*fp1);

                    if(ch==EOF)

                    {

                    cout<<endl<<'\t'<<"Your File has been decrypted successfully"<<endl;

                    return;

                    }

                    else
                    {
                    ch-=k;                              //perform same operation on every character
                    fputc(ch,*fp2);                       // write character to new file
                    curr_position=ftell(*fp1);          // store current position of file pointer
                    percent=(i*max_size)/100;           //calculate percentage
                    if(curr_position==percent)
                    {
                    cout<<'\t'<<'\t'<<i<<" % data decryption completed.."<<endl;
                    i+=10;
                    }
                    }
          }
}
else
   cout<<'\t'<<"Password not matched!!"<<endl;
}

int main(int argc,char *argv[])
{
FILE *fp1,*fp2;
char file_name[100];
intkey,opt,i=1,max_size;
char ch;
do
{
cout<<"                MENU"<<endl;
cout<<"    Option 1.Encrypt a file"<<endl;
```

```
cout<<"    Option 2.Decrypt a file"<<endl;
cout<<"    Enter your Option:"<<endl;
cin>>opt;

switch(opt)
{
case 1:
fp1=fopen(argv[1],"r");                  //open source file
if(fp1==NULL)
{
cout<<'\t'<<'\t'<<argv[1]<<"Error!! File can't be open"<<endl;
exit(1);  //exit in case of error
}
else
{
cout<<'\t'<<argv[1]<<" file found"<<endl;
max_size=file_size(&fp1);
//check for maximum size and store size in  variable
}
cout<<endl<<'\t'<<"Enter encrypted file name( ABC.ecrypt )"<<endl;
fflush(stdin);
// function to empty  memory buffer
cin.get(file_name,50);              // read file name entered by user
fp2=fopen(file_name,"w");       //create new file and open it in write mode

if(fp2!=NULL)
{
cout<<'\t'<<file_name<<" File Created Successfully"<<endl;
}
cout<<'\t'<<"Enter password for encrypted File:: "<<endl;
fflush(stdin);
cin>>key;               //provide a key
encrypt_file(&fp1,&fp2,key,max_size);
fclose(fp1);
fclose(fp2);    //close all file
cout<<'\t'<<"Press Enter to exit"<<endl;
getch();
exit(0);        //end of program

case 2:
fp1=fopen(argv[1],"r");                // open encrypted file in read mode
cout<<'\t'<<"Enter decrypted file name( ABC.decrypt)"<<endl;
fflush(stdin);
cin.get(file_name,50);          //create new decrypted file and open it in read mode
fp2=fopen(file_name,"w");
if(fp1==NULL)
{
cout<<'\t'<<'\t'<<argv[1]<<"Error!! File can't be open"<<endl;
exit(1); //exit in case of error
}
else
{
cout<<'\t'<<argv[1]<<"file found"<<endl;  max_size=file_size(&fp1);
//calculate max size of file and store the size in max_size
}
if(fp2==NULL)
```

```
{
cout<<'\t'<<'\t'<<file_name<<"File cannot be created!!"<<endl;
exit(1);
}
else
cout<<'\t'<<file_name<<" file created successfully"<<endl;
decrypt_file(&fp1,&fp2,max_size);
 ///call decryption function
fclose(fp1);
fclose(fp2);
cout<<'\t'<<"Press Enter To Exit"<<endl;
getch();
exit(0);
default:
cout<<'\t'<<"Please Enter a Valid Option"<<endl;
}
}while(toupper(ch)=='Y');  /// end of while loop if not y
getch();
}
```

REFERENCE

1. www.youtube.com/codewithharry
2. C++ with Saurabh Sukla/youtube.com
3. Let us C by Yashwant Kanetkar.
4. Books on C++ and C languages
5. Geeks for geeks.com
6. WIKIPEDIA
7. GOOGLE