St. Joseph’s Co-Ed School

SUB: COMPUTER SCIENCE (083)

CLASS XII

SESSION 2018-19

A PROJECT

ON

CRYPTOGRAPHY

Guided By: Submitted By:

Mrs. Harpreet Khurana Sarthak Rohatgi Roll No. – 12’46’C

Certificate

This is to certify that the project developed by Sarthak Rohatgi of Class XII entitled Cryptography, is bonafide work of the student in the Computer Science Lab during the academic year 2018-2019 is carried out with the consultation of Teacher Incharge.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Examiner’s Signature

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Principal’s Signature Teacher Incharge

Acknowledgement

It is with great pleasure that I find myself penning down these lines to express my sincere thanks to various people who helped me a long way in completing this project.

The harmonious climate in our school provided proper guide for preparing the project. It was a privilege to have been guided by our computer teacher.

Thanks to all my classmates who helped me during development of this project with their constructive criticism and advice.

Contents

1. Abstract 1-2
2. Problem Description 3
3. Introduction to C++ 4-7
4. Header files used 8
5. System requirements 9
6. Flowchart 10-11
7. Source code 12-29
8. Output 30-34
9. Conclusions & Future Enhancements 35-36
10. Bibliography 37

Abstract

Cryptography is the science of information security. The word is derived from the Greek  **kryptos**, meaning hidden. Cryptography includes techniques such as microdots, merging words with images, and other ways to hide information in storage or transit. Modern cryptography intersects the disciplines of mathematics, computer science, and electrical engineering. Applications of cryptography include ATM cards, computer passwords, and electronic commerce.  
Cryptology prior to the modern age was almost synonymous with *encryption*, the conversion of information from a readable state to apparent nonsense. The sender retained the ability to decrypt the information and therefore avoid unwanted persons being able to read it. Since WWI and the advent of the computer, the methods used to carry out cryptology have become increasingly complex and its application more widespread.  
Modern cryptography follows a strongly scientific approach, and designs cryptographic algorithms around computational hardness assumptions, making such algorithms hard to break by an adversary. Such systems are not unbreakable in theory but it is infeasible to do so by any practical means. These schemes are therefore computationally secure. There exist secure schemes that provably cannot be broken--an example is the one-time pad--but these schemes are more difficult to implement than the theoretically breakable but computationally secure mechanisms.

In this project C++ language is used to maintain all the data. It provides many features like file handling , data can be easily maintained and many features that are required while doing a project.

At present there are many encryption and decryption, especially in the communication system provided in a variety of application.

Problem Description

Encryption and decryption is particularly impacted in the field of military communications and reliable security data to protection for transmitting.

This ciphertext is used in the military is to send information such as direction, strategy, secret codes and other information that can not be know by the national foe during the war.

Creating this system is the process by which information can not be detect by the national foe. The information can not be recognizing by the national foe because this system is using the process to encrypt data and decrypt data.

Introduction to C++

The C++ programming language was developed at AT & T bell laboratories in the early 1980’s by Bjarne Stroustrup. He found ‘C’ lacking for stimulations and decided to extend the language by adding features from Simula 67 which was one of the earliest object oriented languages. The name C++ was coined by Rick Mascitti where ‘++’ is the ‘C’ increment operator. Ever since its birth, C++ involved coping with problems encountered by users and through discussions at AT & T. However, the maturation of the language is attested to by two events:

1. Formation of ANSI committee
2. Publication of “ The Annotated C++ Reference Manual “ by Ellis & Stroustrup

FEATURES OF C++ PROGRAMS

Every C++ program must have a function named main (). Program execution begins at main () and continues by sequentially executing the statements within main (). A program terminates normally following execution of the last statement of main (). Curly braces ({&}) is used to express grouping of statements in C++. Every executable statement in C++ must be terminated by a semicolon (;).

OOP CONCEPTS IN C++

The Object Oriented Programming has been developed with a view to overcome the drawbacks of the conventional programming approaches. The OOP approach is based on certain concepts that help it attain its goal of overcoming the drawbacks or shortcomings of conventional programming approaches. These general concepts of OOP are given below:

DATA ABSTRACTION

It refers to the act of representing essential features without including background details or explanations.

ENCAPSULATION

It is wrapping up of data and functions into a single unit.

MODULARITY

It is the act of partitioning a program into individual components.

INHERITANCE

It is the capability of one class of things to inherit capability or properties from another class.

POLYMORPHISM

It is the ability for a message to be processed in more than one form.

SOME IMPORTANT DATA TYPES IN C++

ARRAYS

Arrays refer to a named list of a finite number n of similar data elements. They maybe 1-dimensional, 2-dimensional or multi-dimensional.

Example: float A[30];

FUNCTION

It is a named part of program, which can be invoked from other parts of the program as often needed.

POINTER

It is a variable which holds a memory address. This address is usually the location of another variable in memory.

Example: char \*ptr;

CLASS

It represents a group of similar objects. A class describes all the properties of a data type and an object is an entity created according to that description.

STRUCTURE

It is a collection of different data types referenced under one name, providing convenient means of keeping related information together.

Header Files Used

* constream.h
  + cin()
  + cout()
  + getch()
  + clrscr()
* fstream.h
  + open()
  + close()
  + ifstream()
  + ofstream()
* process.h
  + exit()
* stdio.h
  + puts()
  + gets()
* windows.h
  + GetSystemMetrics()
  + System(“CLS”)
  + System(“PAUSE”)
  + GetAsyncKeyState()
* graphics.h
  + initwindow();
  + closegraph();
  + outtextxy();
  + line();
  + rectange();
  + cleardevice();
  + settextstyle();
  + kbhit();

System Requirements

SOFTWARE REQUIREMENTS:

Platform: Windows 98/XP/2000.

Front end: GCC\G++ Compiler

Back end: DOS file support.

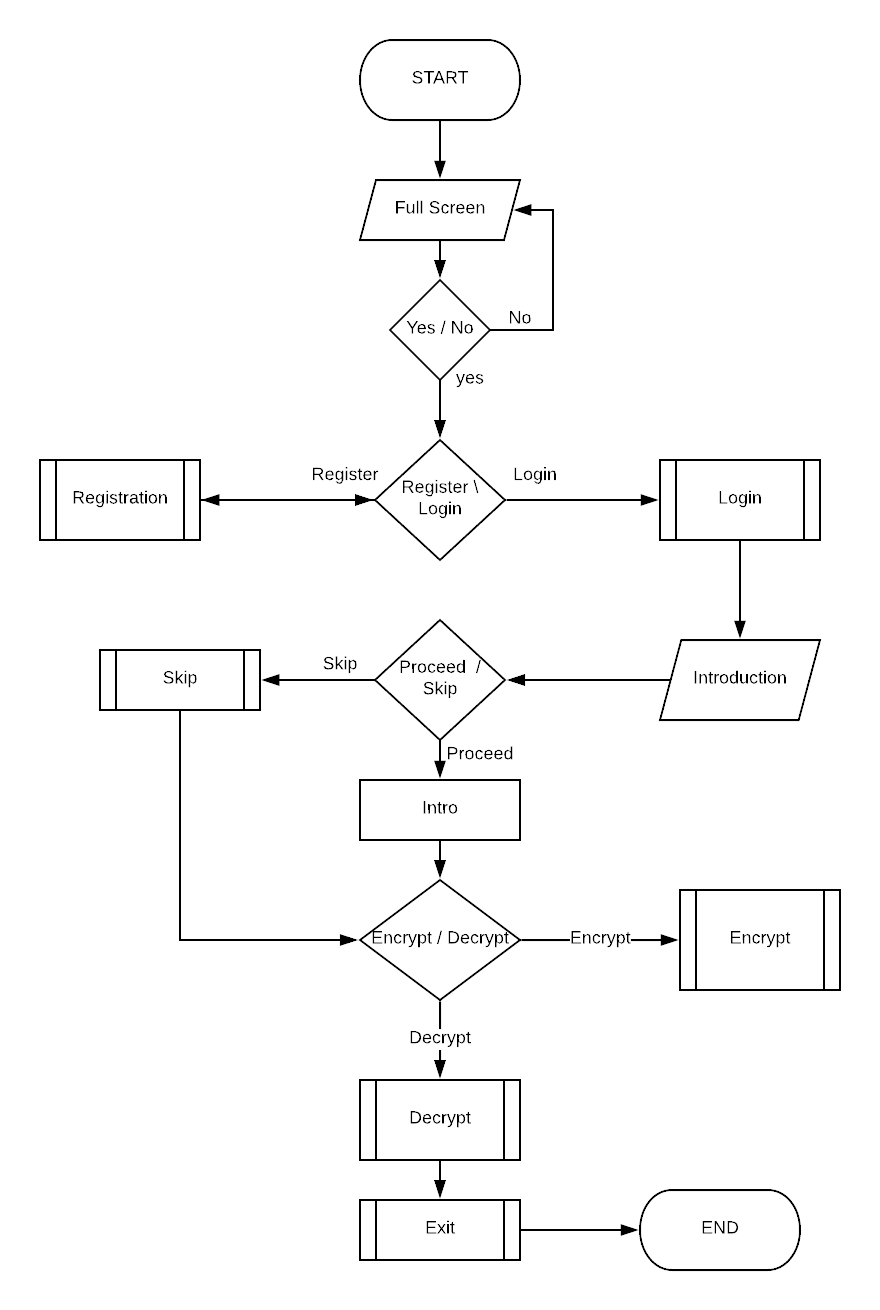
HARDWARE REQUIREMENTS:

PIII 500 MHz or higher.

RAM 64 MB or more.

HDD capacity 2GB or more.

Flowchart



Source Code

//! Pre-processor Commands

#define OR ||

//! Libraries

#include <iostream>

#include <graphics.h>

#include <conio.h>

#include <string.h>

#include <windows.h>

#include <stdio.h>

//! User-Defined Header Files

#include "welcomeScreen.h"

#include "r\_l\_screen.h"

#include "log\_regs.h"

#include "introduction.h"

//! User Defined Functions Declarations

void windowSize(std::string);

DWORD getScreenWidth();

DWORD getScreenHeight();

//! Main Program

int main()

{

std::string getWindowSize;

std::cout << "Question : Do you want this program to run full screen?"

<< std::endl

<< "Answer : ";

windowSize(getWindowSize);

return 0;

}

//! User Defined Functions Definitions

void windowSize(std::string getWindowSize) {

std::cin >> getWindowSize;

system("CLS");

if (

(getWindowSize == "y") OR

(getWindowSize == "Y") OR

(getWindowSize == "ys") OR

(getWindowSize == "YS") OR

(getWindowSize == "Ys") OR

(getWindowSize == "yS") OR

(getWindowSize == "yes") OR

(getWindowSize == "Yes") OR

(getWindowSize == "YEs") OR

(getWindowSize == "YES") OR

(getWindowSize == "yEs") OR

(getWindowSize == "yES") OR

(getWindowSize == "yeS")

)

{

//! To get the screen pixel count

DWORD screenWidth = getScreenWidth();

DWORD screenHeight = getScreenHeight();

initwindow(screenWidth, screenHeight, "", -3, -3);

projectTitle(); //! To call function "projectTitle" from welcomeScreen.h

getch();

closegraph();

}

else {

std::cout << std:: endl << "This program is only compatible for fullscreen mode only" << std::endl << std::endl;

main();

}

}

DWORD getScreenWidth() {

//! To get the screen pixel count of x-axis

DWORD screenWidth = GetSystemMetrics(SM\_CXSCREEN);

return screenWidth;

}

DWORD getScreenHeight() {

//! To get the screen pixel count of y-axis

DWORD screenHeight = GetSystemMetrics(SM\_CYSCREEN);

return screenHeight;

}

#include <iostream>

#include <graphics.h>

#include "welcomeScreen.h"

#include "r\_l\_screen.h"

#include "exitfunc.h"

void projectTitle() {

//! To get the screen pixel count

DWORD screenWidth = getScreenWidth();

DWORD screenHeight = getScreenHeight();

//! Delay of 1 second

delay(1000);

//! For the text, "CRYPTOGRAPHY"

settextstyle(GOTHIC\_FONT, HORIZ\_DIR, 6.7);

int textWidth1 = textwidth("CRYPTOGRAPHY");

int textHeight1 = textheight("CRYPTOGRAPHY");

outtextxy(

(screenWidth - textWidth1)/2,

(screenHeight - (textHeight1+70)) / 2,

"CRYPTOGRAPHY"

);

//! For top and bottom line of text, "CRYPTOGRAPHY"

line(

((screenWidth - textWidth1)/2),

(((screenHeight - textHeight1)/2) - 40),

(screenWidth + textWidth1)/2,

(((screenHeight - textHeight1)/2) - 40)

);

line(

((screenWidth - textWidth1)/2),

(((screenHeight - textHeight1)/2) + 20),

(screenWidth + textWidth1)/2,

(((screenHeight - textHeight1)/2) + 20)

);

//! Delay of 1 second

delay(1000);

//! For the text, "A PROJECET BY"

settextstyle(DEFAULT\_FONT, HORIZ\_DIR, 2);

int textWidth2 = textwidth("A PROJECT BY");

outtextxy(

(screenWidth - textWidth2)/2,

(((screenHeight - textHeight1)/2) + 55),

"A PROJECT BY"

);

//! For the text, "SARTHAK ROHATGI"

settextstyle(BOLD\_FONT, HORIZ\_DIR, 3);

int textWidth3 = textwidth("SARTHAK ROHATGI");

outtextxy(

(screenWidth - textWidth3)/2,

(((screenHeight - textHeight1)/2) + 75),

"SARTHAK ROHATGI"

);

//! For the text, "PRESS ANY KEY TO CONTINUE"

settextstyle(BOLD\_FONT, HORIZ\_DIR, 2);

int textWidth4 = textwidth("PRESS ANY KEY TO CONTINUE");

outtextxy(

(screenWidth - textWidth4)/2,

screenHeight - 100,

"PRESS ANY KEY TO CONTINUE"

);

getch();

cleardevice();

setup(); //! To call function "setup" from r\_l\_screen.h

}

//! Pre-processor Commands

#define AND &&

//! Libraries & Header Files

#include <iostream>

#include <graphics.h>

#include "r\_l\_screen.h"

#include "welcomeScreen.h"

#include "log\_regs.h"

#include "encryptDecrypt.h"

#include "exitfunc.h"

POINT cursorPosition;

//! To get the screen Metrics x-axis & y-axis

DWORD screenWidth = getScreenWidth();

DWORD screenHeight = getScreenHeight();

//! Back Button

void backbuttn(){

settextstyle(BOLD\_FONT, HORIZ\_DIR, 2);

outtextxy(((screenWidth\*4)/100), (screenHeight-((screenWidth\*6)/100)), "BACK");

int textHeight = textheight("BACK");

int textWidth = textwidth("BACK");

rectangle(

((screenWidth\*4)/100) - 10,

(screenHeight-((screenWidth\*6)/100)) - 5,

((screenWidth\*4)/100) + textWidth + 10,

(screenHeight-((screenWidth\*6)/100)) + textHeight + 5

);

}

//! Exit Button

void exitbuttn(){

settextstyle(BOLD\_FONT, HORIZ\_DIR, 2);

int textHeight = textheight("EXIT");

int textWidth = textwidth("EXIT");

outtextxy((screenWidth - textWidth - 25), (screenHeight-((screenWidth\*6)/100)), "EXIT");

rectangle(

((screenWidth - textWidth - 25) - 10),

((screenHeight-((screenWidth\*6)/100)) - 5),

((screenWidth - textWidth + 25) + 10) ,

((screenHeight-((screenWidth\*6)/100)) + 5 + textHeight)

);

}

void setup() {

//! Delay of .35 second

delay(350);

//! For the text "WELCOME" & line below it

settextstyle(GOTHIC\_FONT, HORIZ\_DIR, 7);

int textHeight1 = textheight("WELCOME");

int textWidth1 = textwidth("WELCOME");

outtextxy(

((screenWidth - textWidth1)/2),

((screenHeight/4) - textHeight1/2),

"WELCOME"

);

line(

((screenWidth - textWidth1)/2),

(((screenHeight/4) + (textHeight1)/2)),

((screenWidth + textWidth1)/2),

(((screenHeight/4) + (textHeight1)/2))

);

//! For the text "REGISTER" & "LOGIN"

settextstyle(COMPLEX\_FONT, HORIZ\_DIR, 5);

int textHeight2 = textheight("REGISTER");

int textWidth2 = textwidth("REGISTER");

int textHeight3 = textheight("LOGIN");

int textWidth3 = textwidth("LOGIN");

outtextxy(

((screenWidth - textWidth2)/2),

(screenHeight/2),

"REGISTER"

);

outtextxy(

((screenWidth - textWidth3)/2),

((screenHeight/2) + textHeight1\*2),

"LOGIN"

);

//! For the text "BACK"

settextstyle(BOLD\_FONT, HORIZ\_DIR, 2);

int textHeight4 = textheight("BACK");

int textWidth4 = textwidth("BACK");

//! For the text "EXIT"

settextstyle(BOLD\_FONT, HORIZ\_DIR, 2);

int textHeight5 = textheight("EXIT");

int textWidth5 = textwidth("EXIT");

backbuttn();

exitbuttn();

while(1) {

GetCursorPos(&cursorPosition);

//! For Registration

if((GetAsyncKeyState(VK\_LBUTTON)) AND

(cursorPosition.x > (screenWidth - textWidth2)/2) AND

(cursorPosition.x < ((screenWidth - textWidth2)/2)+textWidth2) AND

(cursorPosition.y > (screenHeight/2)) AND

(cursorPosition.y < (screenHeight/2)+textHeight2)

){

cleardevice();

regs\_screen();

}

//! For Login

if((GetAsyncKeyState(VK\_LBUTTON)) AND

(cursorPosition.x > (screenWidth - textWidth3)/2) AND

(cursorPosition.x < ((screenWidth - textWidth3)/2) + textWidth3) AND

(cursorPosition.y > (screenHeight/2) + textHeight1\*2) AND

(cursorPosition.y < ((screenHeight/2) + textHeight1\*2) + textHeight3)

){

cleardevice();

logs\_screen();

}

//! For back button

if((GetAsyncKeyState(VK\_LBUTTON)) AND

(cursorPosition.x > ((screenWidth\*4)/100) - 10) AND

(cursorPosition.x < (((screenWidth\*4)/100) + textWidth4 + 10)) AND

(cursorPosition.y > (screenHeight-((screenWidth\*6)/100)) - 5) AND

(cursorPosition.y < (screenHeight-((screenWidth\*6)/100)) + textHeight4 + 5)

){

cleardevice();

projectTitle();

}

//! For exit button

if((GetAsyncKeyState(VK\_LBUTTON)) AND

(cursorPosition.x > ((screenWidth - textWidth5 - 25) - 10)) AND

(cursorPosition.x < ((screenWidth - textWidth5 + 25) + 10)) AND

(cursorPosition.y > ((screenHeight-((screenWidth\*6)/100)) - 5)) AND

(cursorPosition.y < ((screenHeight-((screenWidth\*6)/100)) + 5 + textHeight5))

){

exitfunc();

}

}

}

#include <iostream>

#include <fstream>

#include <istream>

#include <stdio.h>

#include <graphics.h>

#include <string>

#include <windows.h>

#include "r\_l\_screen.h"

#include "welcomeScreen.h"

#include "log\_regs.h"

#include "introduction.h"

#include "exitfunc.h"

void proceedbuttn();

void regs\_screen() {

DWORD screenWidth = getScreenWidth();

DWORD screenHeight = getScreenHeight();

//! For the text "REGISTRATION" & line below it

settextstyle(GOTHIC\_FONT, HORIZ\_DIR, 7);

int textHeightRegistration = textheight("REGISTRATION");

int textWidthRegistration = textwidth("REGISTRATION");

outtextxy(

((screenWidth - textWidthRegistration)/2),

((screenHeight/4) - textHeightRegistration/2),

"REGISTRATION"

);

line(

((screenWidth - textWidthRegistration)/2),

(((screenHeight/4) + (textHeightRegistration)/2)),

((screenWidth + textWidthRegistration)/2),

(((screenHeight/4) + (textHeightRegistration)/2))

);

//! For the text1

settextstyle(DEFAULT\_FONT, HORIZ\_DIR, 2);

int textWidthText1 = textwidth("Click on \"PROCEED\" below to REGISTER" );

outtextxy(

(screenWidth - textWidthText1)/2,

((screenHeight/3) + 55),

"Click on \"PROCEED\" below to REGISTER"

);

//! For the text2

settextstyle(DEFAULT\_FONT, HORIZ\_DIR, 2);

int textWidthText2 = textwidth("where you will be asked for credentials." );

outtextxy(

(screenWidth - textWidthText2)/2,

((screenHeight/3) + 95),

"where you will be asked for credentials."

);

//! For the text "BACK"

settextstyle(BOLD\_FONT, HORIZ\_DIR, 2);

int textHeight1 = textheight("BACK");

int textWidth1 = textwidth("BACK");

//! For the text "PROCEED"

settextstyle(BOLD\_FONT, HORIZ\_DIR, 2);

int textHeightProceed = textheight("PROCEED");

int textWidthProceed = textwidth("PROCEED");

backbuttn();

proceedbuttn();

while(1) {

POINT cursorPosition;

GetCursorPos(&cursorPosition);

if((GetAsyncKeyState(VK\_LBUTTON)) &&

(cursorPosition.x > ((screenWidth\*4)/100) - 10) &&

(cursorPosition.x < (((screenWidth\*4)/100) + textWidth1 + 10)) &&

(cursorPosition.y > (screenHeight-((screenWidth\*6)/100)) - 5) &&

(cursorPosition.y < (screenHeight-((screenWidth\*6)/100)) + textHeight1 + 5)

){

cleardevice();

setup();

}

else if ((GetAsyncKeyState(VK\_LBUTTON)) &&

(cursorPosition.x > (((screenWidth - textWidthProceed)/2) - 10)) &&

(cursorPosition.x < (((screenWidth + textWidthProceed)/2) + 10)) &&

(cursorPosition.y > (((screenHeight/2)+ 100) - 5)) &&

(cursorPosition.y < (((screenHeight/2) + textHeightProceed + 100) + 5))

){

delay(500);

closegraph();

regs();

}

}

}

void logs\_screen() {

DWORD screenWidth = getScreenWidth();

DWORD screenHeight = getScreenHeight();

//! For the text "LOGIN" & line below it

settextstyle(GOTHIC\_FONT, HORIZ\_DIR, 7);

int textHeightLogin = textheight("LOGIN");

int textWidthLogin = textwidth("LOGIN");

outtextxy(

((screenWidth - textWidthLogin)/2),

((screenHeight/4) - textHeightLogin/2),

"LOGIN"

);

line(

((screenWidth - textWidthLogin)/2),

(((screenHeight/4) + (textHeightLogin)/2)),

((screenWidth + textWidthLogin)/2),

(((screenHeight/4) + (textHeightLogin)/2))

);

//! For the text1

settextstyle(DEFAULT\_FONT, HORIZ\_DIR, 2);

int textWidthText1 = textwidth("Click on \"PROCEED\" below to LOGIN" );

outtextxy(

(screenWidth - textWidthText1)/2,

((screenHeight/3) + 55),

"Click on \"PROCEED\" below to LOGIN"

);

//! For the text2

settextstyle(DEFAULT\_FONT, HORIZ\_DIR, 2);

int textWidthText2 = textwidth("where you will be asked for credentials." );

outtextxy(

(screenWidth - textWidthText2)/2,

((screenHeight/3) + 95),

"where you will be asked for credentials."

);

//! For the text "BACK"

settextstyle(BOLD\_FONT, HORIZ\_DIR, 2);

int textHeight1 = textheight("BACK");

int textWidth1 = textwidth("BACK");

//! For the text "PROCEED"

settextstyle(BOLD\_FONT, HORIZ\_DIR, 2);

int textHeightProceed = textheight("PROCEED");

int textWidthProceed = textwidth("PROCEED");

backbuttn();

proceedbuttn();

while(1) {

POINT cursorPosition;

GetCursorPos(&cursorPosition);

//! For back button

if((GetAsyncKeyState(VK\_LBUTTON)) &&

(cursorPosition.x > ((screenWidth\*4)/100) - 10) &&

(cursorPosition.x < (((screenWidth\*4)/100) + textWidth1 + 10)) &&

(cursorPosition.y > (screenHeight-((screenWidth\*6)/100)) - 5) &&

(cursorPosition.y < (screenHeight-((screenWidth\*6)/100)) + textHeight1 + 5)

){

cleardevice();

setup();

}

//! For proceed button

else if ((GetAsyncKeyState(VK\_LBUTTON)) &&

(cursorPosition.x > (((screenWidth - textWidthProceed)/2) - 10)) &&

(cursorPosition.x < (((screenWidth + textWidthProceed)/2) + 10)) &&

(cursorPosition.y > (((screenHeight/2)+ 100) - 5)) &&

(cursorPosition.y < (((screenHeight/2) + textHeightProceed + 100) + 5))

){

delay(500);

closegraph();

logs();

}

}

}

void regs() {

//! To get the screen pixel count

DWORD screenWidth = getScreenWidth();

DWORD screenHeight = getScreenHeight();

std::string reguser, regpass, regfname, reglname;

std::cout << std::endl << std::endl << "\t\t\t\t REGISTRATION" << std::endl << std::endl;

std::cout << std::endl << "Enter your first name - ";

std::cin >> regfname;

std::cout << std::endl << "Enter your last name - ";

std::cin >> reglname;

std::cout << std::endl << "Create your Username - ";

std::cin >> reguser;

std::cout << std::endl << "Create your Password - ";

std::cin >> regpass;

std::ofstream reg("data.txt", std::ios::app);

reg << regfname << "\t\t\t" << reglname << "\t\t\t" << reguser << "\t\t\t" << regpass << std::endl << std::endl;

system("CLS");

for (int i = 0 ; i <= 10 ; i ++) {

std::cout << std::endl;

}

std::cout << "\t Congratulations " << regfname << ". Your Registration is successful!";

delay(2000);

system("CLS");

initwindow(screenWidth, screenHeight, "", -3, -3);

setup();

}

void logs() {

//! To get the screen pixel count

DWORD screenWidth = getScreenWidth();

DWORD screenHeight = getScreenHeight();

bool exist;

int c;

std::string user, pass, u, p;

system("CLS");

std::cout << "Enter Username - ";

std::cin >> user;

std::cout << std::endl;

std::cout << "Enter Password - ";

std::cin >> pass;

std::ifstream input("data.txt");

while(input >> u >> p){

if(u != user || p!= pass) {

exist = false;

}

else if(u == user && p == pass) {

exist = true;

}

}

if(exist == true) {

system("CLS");

for (int i = 0 ; i <= 10 ; i ++) {

std::cout << std::endl;

}

std::cout << "\t Login Successful \"" << user << "\". You are being redirected.";

delay(2000);

system("CLS");

initwindow(screenWidth, screenHeight, "", -3, -3);

intro();

}

else if(exist == false) {

system("CLS");

for (int i = 0 ; i <= 10 ; i ++) {

std::cout << std::endl;

}

std::cout << "\t\t\tIncorrect username or password" << std::endl << std:: endl<< "\t\t\tTry again";

delay(2000);

logs();

}

}

//! Proceed Button

void proceedbuttn(){

DWORD screenWidth = getScreenWidth();

DWORD screenHeight = getScreenHeight();

settextstyle(BOLD\_FONT, HORIZ\_DIR, 2);

int textHeightProceed = textheight("PROCEED");

int textWidthProceed = textwidth("PROCEED");

outtextxy(((screenWidth - textWidthProceed)/2), ((screenHeight)/2)+ 100, "PROCEED");

rectangle(

((screenWidth - textWidthProceed)/2) - 10,

(((screenHeight)/2)+ 100) - 5,

((screenWidth + textWidthProceed)/2) + 10,

(((screenHeight)/2) + textHeightProceed + 100) + 5

);

}

//! Libraries & Header Files

#include <iostream>

#include <graphics.h>

#include "welcomeScreen.h"

#include "r\_l\_screen.h"

#include "log\_regs.h"

#include "introduction.h"

#include "encryptDecrypt.h"

#include "exitfunc.h"

void intro() {

//!To get the screen pixel count of x-axis & y-axis

DWORD screenWidth = getScreenWidth();

DWORD screenHeight = getScreenHeight();

//! For the text "INTRODUCTION" & line below it

settextstyle(GOTHIC\_FONT, HORIZ\_DIR, 7);

int textHeightIntroduction = textheight("INTRODUCTION");

int textWidthIntroduction = textwidth("INTRODUCTION");

outtextxy(

((screenWidth - textWidthIntroduction)/2),

((screenHeight/4) - textHeightIntroduction/2),

"INTRODUCTION"

);

line(

((screenWidth - textWidthIntroduction)/2),

(((screenHeight/4) + (textHeightIntroduction)/2)),

((screenWidth + textWidthIntroduction)/2),

(((screenHeight/4) + (textHeightIntroduction)/2))

);

//! For the text1

settextstyle(DEFAULT\_FONT, HORIZ\_DIR, 2);

int textWidthText1 = textwidth("Click on \"PROCEED\" below to read INTRODUCTION" );

outtextxy(

(screenWidth - textWidthText1)/2,

((screenHeight/3) + 55),

"Click on \"PROCEED\" below to read INTRODUCTION"

);

//! For the text2

settextstyle(DEFAULT\_FONT, HORIZ\_DIR, 2);

int textWidthText2 = textwidth("OR" );

outtextxy(

(screenWidth - textWidthText2)/2,

((screenHeight/3) + 95),

"OR"

);

//! For the text3

settextstyle(DEFAULT\_FONT, HORIZ\_DIR, 2);

int textWidthText3 = textwidth("Click on \"SKIP\" below to skip it." );

outtextxy(

(screenWidth - textWidthText3)/2,

((screenHeight/3) + 135),

"Click on \"SKIP\" below to skip it."

);

//! For the text "PROCEED"

settextstyle(BOLD\_FONT, HORIZ\_DIR, 2);

int textHeightProceed = textheight("PROCEED");

int textWidthProceed = textwidth("PROCEED");

//! For the text "SKIP"

settextstyle(BOLD\_FONT, HORIZ\_DIR, 2);

int textHeightSkip = textheight("PROCEED");

int textWidthSkip = textwidth("PROCEED");

//! For the text "BACK"

settextstyle(BOLD\_FONT, HORIZ\_DIR, 2);

int textHeight1 = textheight("BACK");

int textWidth1 = textwidth("BACK");

//! For the text "EXIT"

settextstyle(BOLD\_FONT, HORIZ\_DIR, 2);

int textHeight5 = textheight("EXIT");

int textWidth5 = textwidth("EXIT");

skip();

proceedbuttn();

backbuttn();

exitbuttn();

while(1) {

POINT cursorPosition;

GetCursorPos(&cursorPosition);

//! For back button

if((GetAsyncKeyState(VK\_LBUTTON)) &&

(cursorPosition.x > ((screenWidth\*4)/100) - 10) &&

(cursorPosition.x < (((screenWidth\*4)/100) + textWidth1 + 10)) &&

(cursorPosition.y > (screenHeight-((screenWidth\*6)/100)) - 5) &&

(cursorPosition.y < (screenHeight-((screenWidth\*6)/100)) + textHeight1 + 5)

){

cleardevice();

setup();

}

//! For exit button

else if((GetAsyncKeyState(VK\_LBUTTON)) &&

(cursorPosition.x > ((screenWidth - textWidth5 - 25) - 10)) &&

(cursorPosition.x < ((screenWidth - textWidth5 + 25) + 10)) &&

(cursorPosition.y > ((screenHeight-((screenWidth\*6)/100)) - 5)) &&

(cursorPosition.y < ((screenHeight-((screenWidth\*6)/100)) + 5 + textHeight5))

){

exitfunc();

}

//! For skip button

else if ((GetAsyncKeyState(VK\_LBUTTON)) &&

(cursorPosition.x > (((screenWidth - textWidthSkip)/2) - 10)) &&

(cursorPosition.x < (((screenWidth + textWidthSkip)/2) + 10)) &&

(cursorPosition.y > (((screenHeight/2)+ 150) - 5)) &&

(cursorPosition.y < (((screenHeight/2) + textHeightSkip + 150) + 5))

){

delay(500);

cleardevice();

encORdec();

}

//! For proceed button

else if ((GetAsyncKeyState(VK\_LBUTTON)) &&

(cursorPosition.x > (((screenWidth - textWidthProceed)/2) - 10)) &&

(cursorPosition.x < (((screenWidth + textWidthProceed)/2) + 10)) &&

(cursorPosition.y > (((screenHeight/2)+ 100) - 5)) &&

(cursorPosition.y < (((screenHeight/2) + textHeightProceed + 100) + 5))

){

delay(500);

closegraph();

std::cout << "The Caesar Cipher technique is one of the earliest and simplest method of encryption technique." << std::endl << std::endl

<< "It\’s simply a type of substitution cipher, i.e., each letter of a given text is replaced by a letter some fixed number of positions down the alphabet. For example with a shift of 1, A would be replaced by B, B would become C, and so on." << std::endl << std::endl

<< "The method is apparently named after Julius Caesar, who apparently used it to communicate with his officials." << std::endl << std::endl;

system("PAUSE");

initwindow(screenWidth, screenHeight, "", -3, -3);

encORdec();

}

}

}

//! skip Button

void skip(){

DWORD screenWidth = getScreenWidth();

DWORD screenHeight = getScreenHeight();

settextstyle(BOLD\_FONT, HORIZ\_DIR, 2);

int textHeightSkip = textheight("SKIP");

int textWidthSkip = textwidth("SKIP");

outtextxy(((screenWidth - textWidthSkip)/2), ((screenHeight)/2)+ 150, "SKIP");

rectangle(

((screenWidth - textWidthSkip)/2) - 10,

(((screenHeight)/2)+ 150) - 5,

((screenWidth + textWidthSkip)/2) + 10,

(((screenHeight)/2) + textHeightSkip + 150) + 5

);

}

//! Libraries & Header Files

#include <iostream>

#include <graphics.h>

#include <stdio.h>

#include "welcomeScreen.h"

#include "r\_l\_screen.h"

#include "log\_regs.h"

#include "introduction.h"

#include "encryptDecrypt.h"

#include "exitfunc.h"

void exitfunc() {

//!To get the screen pixel count of x-axis & y-axis

DWORD screenWidth = getScreenWidth();

DWORD screenHeight = getScreenHeight();

cleardevice();

//! For the text

settextstyle(GOTHIC\_FONT, HORIZ\_DIR, 5);

int textHeightIntroduction = textheight("Thank You for using ");

int textWidthIntroduction = textwidth("Thank You for using ");

outtextxy(

((screenWidth - textWidthIntroduction)/2),

((screenHeight/4) - textHeightIntroduction/2),

"Thank You for using "

);

settextstyle(GOTHIC\_FONT, HORIZ\_DIR, 5);

int textHeightIntroduction1 = textheight("CRYPTOGRAPHY");

int textWidthIntroduction1 = textwidth("CRYPTOGRAPHY");

outtextxy(

((screenWidth - textWidthIntroduction1)/2),

((screenHeight/2)),

"CRYPTOGRAPHY"

);

delay(5000);

closegraph();

exit(0);

}

//! Libraries & Header Files

#include <iostream>

#include <graphics.h>

#include <stdio.h>

#include "welcomeScreen.h"

#include "r\_l\_screen.h"

#include "log\_regs.h"

#include "introduction.h"

#include "encryptDecrypt.h"

#include "exitfunc.h"

void encORdec() {

POINT cursorPosition;

//! To get the screen pixel count

DWORD screenWidth = getScreenWidth();

DWORD screenHeight = getScreenHeight();

//! Delay of .35 second

delay(350);

//! For the text "ENCRYPT/DECRYPT" & line below it

settextstyle(GOTHIC\_FONT, HORIZ\_DIR, 7);

int textHeight1 = textheight("ENCRYPT/DECRYPT");

int textWidth1 = textwidth("ENCRYPT/DECRYPT");

outtextxy(

((screenWidth - textWidth1)/2),

((screenHeight/4) - textHeight1/2),

"ENCRYPT/DECRYPT"

);

line(

((screenWidth - textWidth1)/2),

(((screenHeight/4) + (textHeight1)/2)),

((screenWidth + textWidth1)/2),

(((screenHeight/4) + (textHeight1)/2))

);

//! For the text "ENCRYPT" & "DECRYPT"

settextstyle(COMPLEX\_FONT, HORIZ\_DIR, 5);

int textHeight2 = textheight("ENCRYPT");

int textWidth2 = textwidth("ENCRYPT");

int textHeight3 = textheight("DECRYPT");

int textWidth3 = textwidth("DECRYPT");

outtextxy(

((screenWidth - textWidth2)/2),

(screenHeight/2),

"ENCRYPT"

);

outtextxy(

((screenWidth - textWidth3)/2),

((screenHeight/2) + textHeight1\*2),

"DECRYPT"

);

//! For the text "EXIT"

settextstyle(BOLD\_FONT, HORIZ\_DIR, 2);

int textHeight5 = textheight("EXIT");

int textWidth5 = textwidth("EXIT");

backbuttn();

exitbuttn();

while(1) {

GetCursorPos(&cursorPosition);

//! For ENCRYPT

if((GetAsyncKeyState(VK\_LBUTTON)) &&

(cursorPosition.x > (screenWidth - textWidth2)/2) &&

(cursorPosition.x < ((screenWidth - textWidth2)/2)+textWidth2) &&

(cursorPosition.y > (screenHeight/2)) &&

(cursorPosition.y < (screenHeight/2)+textHeight2)

){

closegraph();

encrypt();

}

//! For DECRYPT

if((GetAsyncKeyState(VK\_LBUTTON)) &&

(cursorPosition.x > (screenWidth - textWidth3)/2) &&

(cursorPosition.x < ((screenWidth - textWidth3)/2) + textWidth3) &&

(cursorPosition.y > (screenHeight/2) + textHeight1\*2) &&

(cursorPosition.y < ((screenHeight/2) + textHeight1\*2) + textHeight3)

){

closegraph();

decrypt();

}

//! For exit button

if((GetAsyncKeyState(VK\_LBUTTON)) &&

(cursorPosition.x > ((screenWidth - textWidth5 - 25) - 10)) &&

(cursorPosition.x < ((screenWidth - textWidth5 + 25) + 10)) &&

(cursorPosition.y > ((screenHeight-((screenWidth\*6)/100)) - 5)) &&

(cursorPosition.y < ((screenHeight-((screenWidth\*6)/100)) + 5 + textHeight5))

){

exitfunc();

}

}

}

void encrypt() {

//! To get the screen pixel count

DWORD screenWidth = getScreenWidth();

DWORD screenHeight = getScreenHeight();

system("CLS");

std::string choice;

char message[100], ch;

int i, key;

std::cout << "\t\t\t\t ENCRYPTION" << std::endl << std::endl;

std::cout << "Enter a message to encrypt: ";

std::cin.ignore();

std::cin.getline(message, 100);

std::cout << "Shift key: ";

std::cin >> key;

for(i = 0; message[i] != '\0'; ++i){

ch = message[i];

if(ch >= 'a' && ch <= 'z'){

ch = ch + key;

if(ch > 'z'){

ch = ch - 'z' + 'a' - 1;

}

message[i] = ch;

}

else if(ch >= 'A' && ch <= 'Z'){

ch = ch + key;

if(ch > 'Z'){

ch = ch - 'Z' + 'A' - 1;

}

message[i] = ch;

}

}

std::cout << "Encrypted message - ";

puts(message);

std::cout << std::endl << std::endl << "Want to encrypt another? (Yes OR No) - ";

std::cin >> choice;

if(choice == "yes"|| choice == "Yes") {

encrypt();

}

else {

initwindow(screenWidth, screenHeight, "", -3, -3);

encORdec();

}

}

void decrypt() {

//! To get the screen pixel count

DWORD screenWidth = getScreenWidth();

DWORD screenHeight = getScreenHeight();

system("CLS");

std::string choice;

char message[100], ch;

int i, key;

std::cout << "Enter a message to decrypt: ";

std::cin.ignore();

std::cin.getline(message, 100);

std::cout << "Shift key: ";

std::cin >> key;

for(i = 0; message[i] != '\0'; ++i){

ch = message[i];

if(ch >= 'a' && ch <= 'z'){

ch = ch - key;

if(ch < 'a'){

ch = ch + 'z' - 'a' + 1;

}

message[i] = ch;

}

else if(ch >= 'A' && ch <= 'Z'){

ch = ch - key;

if(ch > 'a'){

ch = ch + 'Z' - 'A' + 1;

}

message[i] = ch;

}

}

std::cout << "Decrypted message: " << message;

std::cout << std::endl << std::endl << "Want to decrypt another? (Yes OR No) - ";

std::cin >> choice;

if(choice == "yes"|| choice == "Yes") {

decrypt();

}

else {

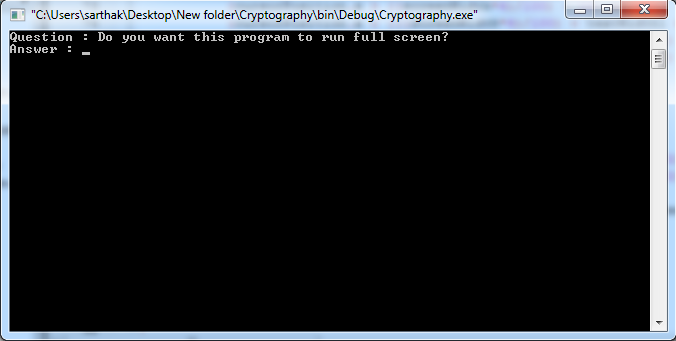
initwindow(screenWidth, screenHeight, "", -3, -3);

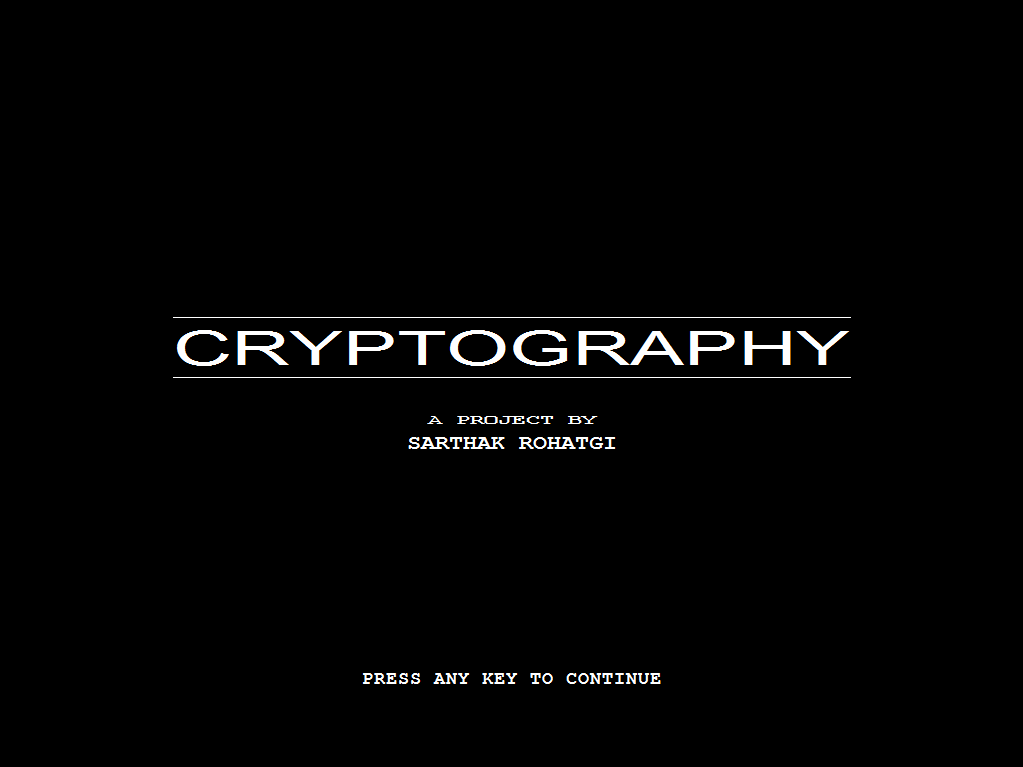
encORdec();

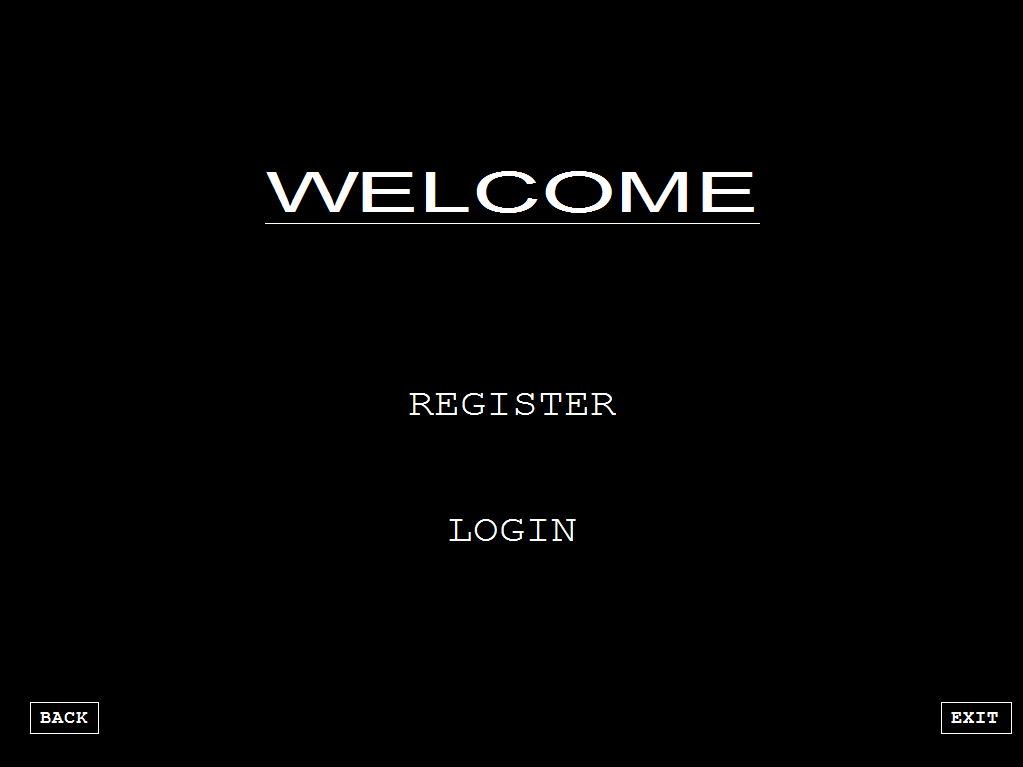
}

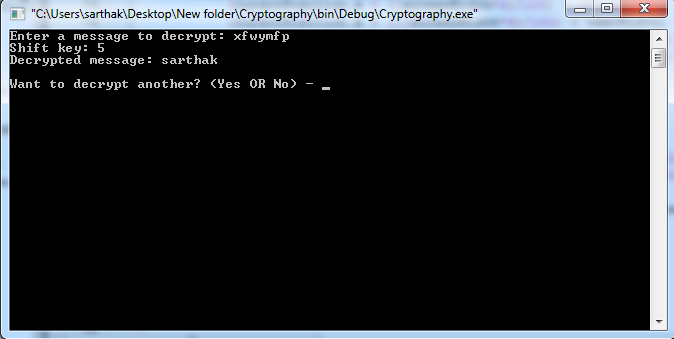
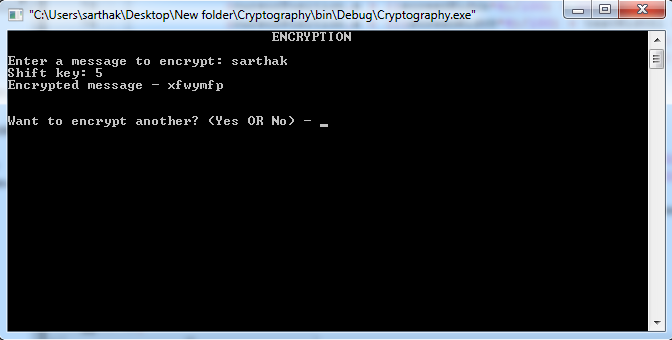
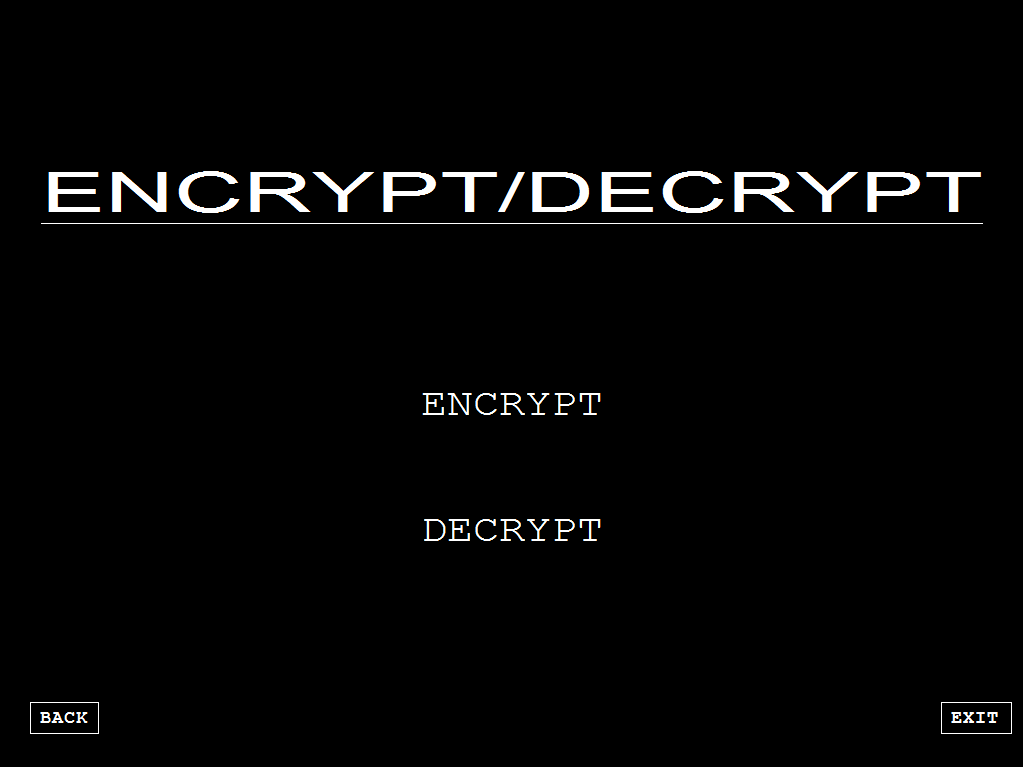
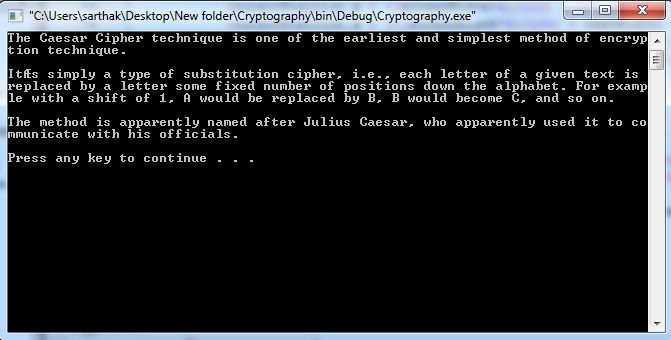
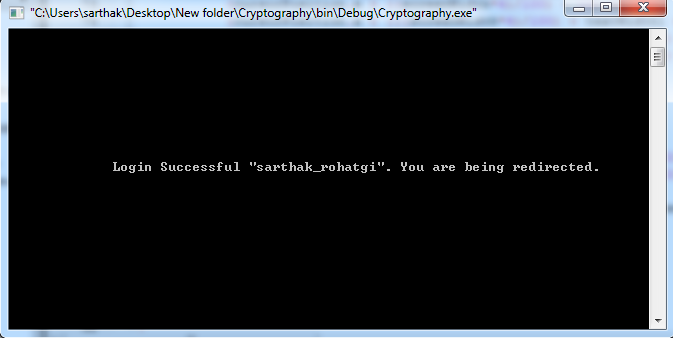
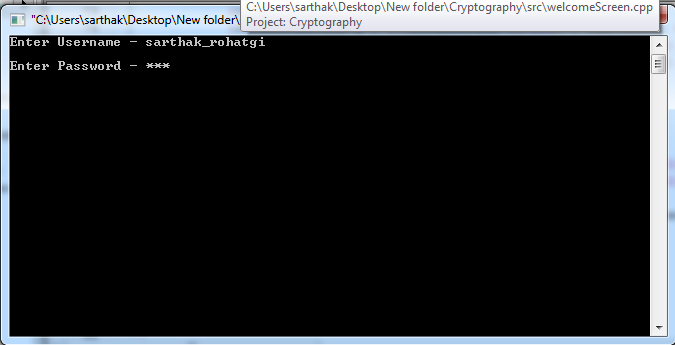
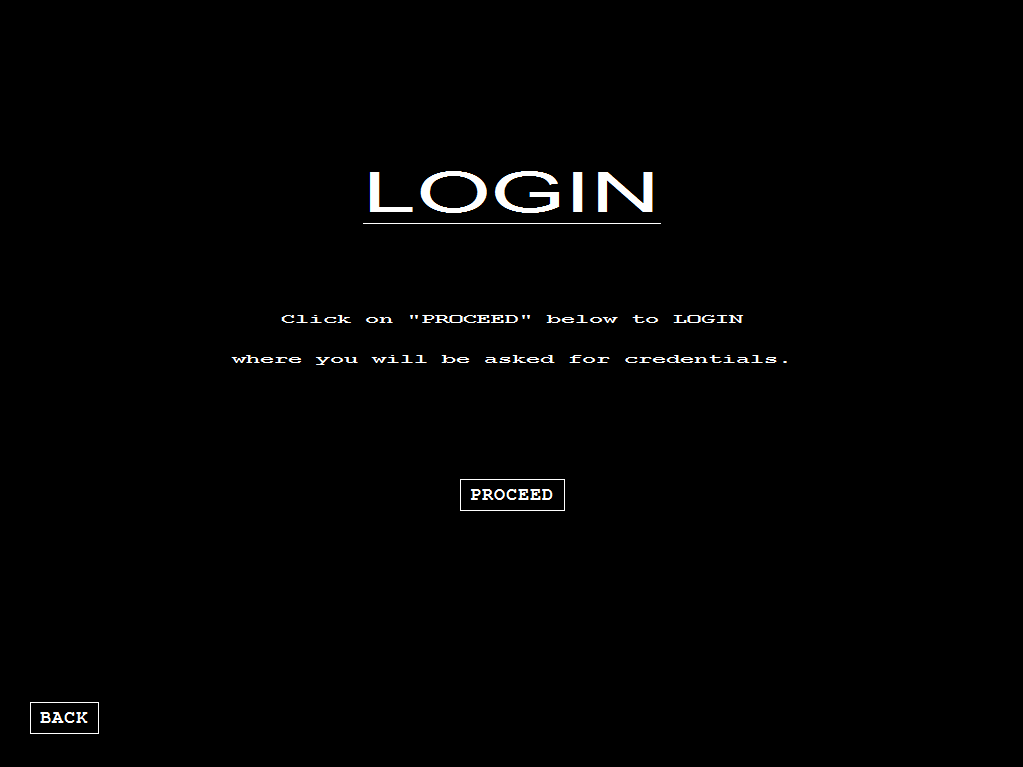
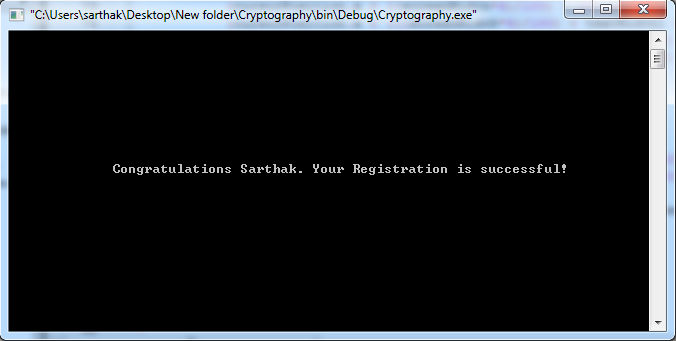
}

Output









Future Enhancements and Conclusion

The conclusion that can be drawn from this project is that this program is meant to enter/edit/modify/delete/search a pre-added record of the customers. And can be used can be used in restaurants.

The enhancement that can be done with this software is that it always asks for the customer’s serial number at the time of searching a record. So, rather than asking only for serial number this program can ask for the name beginning with any particular alphabet/bill of the customer that is given by the user for searching and even more convenient searching options can be added to this. Similarly, for other options like modify, delete other means of searching the records can also be included.

On the other hand, the program can be upgraded to software which can edit multiple records and can add a record of customers on the much greater level i.e. it can be more reliable if this program allows the user (generally the management staff) to enter the name of the entire customer dining in the restaurant and sorted according to the users requirement.

Bibliography

* Computer Science with C++ Textbook for Class XI, by Sumita Arora.
* Computer Science with C++ Textbook for Class XII Volume-1, by Sumita Arora.

37