PROJECT REPORT

OBJECT ORIENTED PROGRAMMING
(CT-260)

PROJECT TITLE : "TEXT ANALYZER "

Group Members :

MUHAMMAD ABDULLAH (CR-22027) (Group Lead)
M. SHEHERYAR AMIR (CR-22008)
SYED GHUFRAN RAZA (CR-22028)

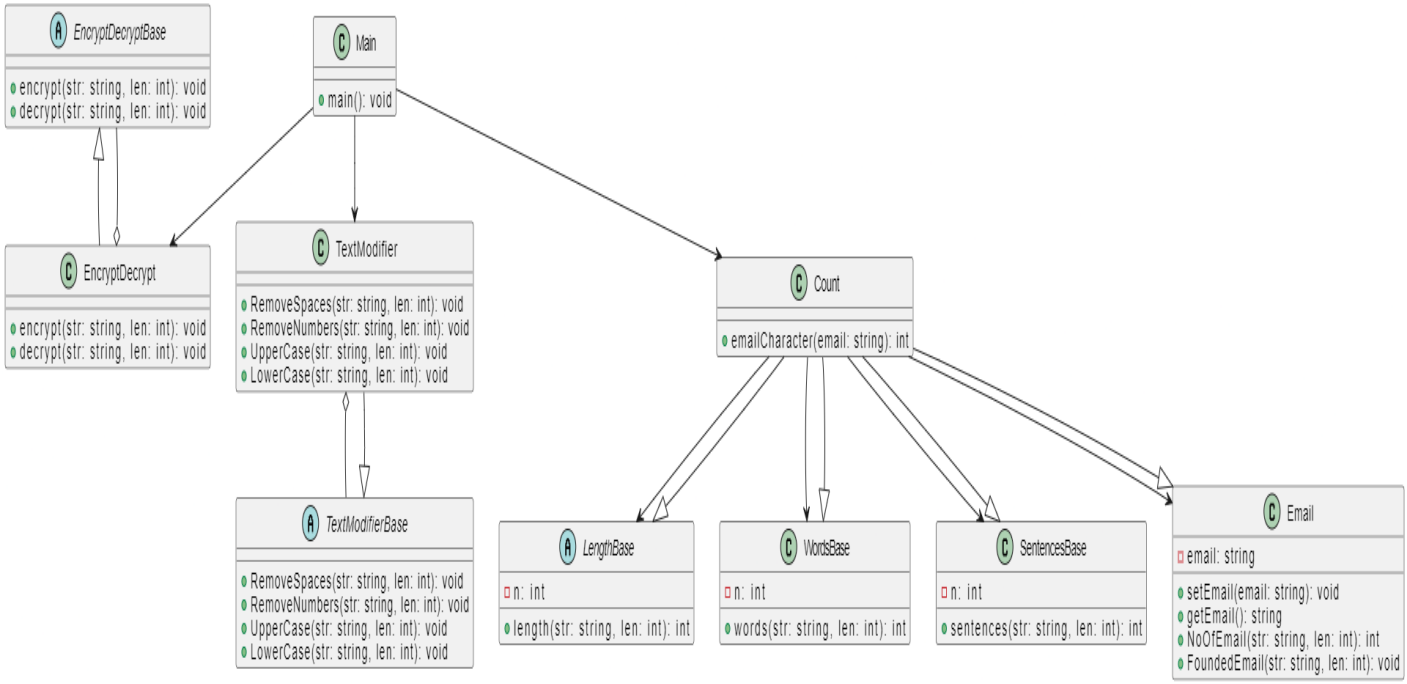
INTRODUCTION:

The designed code is a text analyzer application which is designed to perform various operations on a given text, including encryption, decryption, text modification, and text analysis. It is implemented using Object-Oriented Programming principles to ensure code organization, modularity, and reusability.

The text analyzer application offers a user-friendly interface for interacting with the text and performing different operations. The user is prompted to enter a text, and then the application provides a summary of the text, including the total number of characters, words, and sentences present in the text. Additionally, the application detects and displays any email addresses found in the text.

Overall, the "Text Analyzer" project offers a range of functionalities to analyze and manipulate text, providing users with a comprehensive tool for text analysis and modification.

UML CLASS DIAGRAM:



OUTPUT:

```
---------------------- TEXT ANALYZER ------------------
|                      PROJECT MADE BY:               |
|                  "MUHAMMAD ABDULLAH"(CR-027)        |
|                  "SHEHERYAR AMIR"(CR-008)           |
|                  "SYED GHUFRAN RAZA"(CR-028)        |
-----------------------------------------------------


Enter the Text : Hello, My name is Sarah. My Email is ssarah@gmail.com &      420     my friends mail is ayesha@yahoo.com 6949.         We
 work48 as a Content-Creator          from past 2 years.

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

The Entered Text is : Hello, My name is Sarah. My Email is ssarah@gmail.com &      420     my friends mail is ayesha@yahoo.com 6949.
   We work48 as a Content-Creator          from past 2 years.

       The SUMMARY of Your Text is :
Total Number of Characters : 179

Total Number of Words : 26

Total Number of Sentences : 5

Number OF Founded Emails : 2

Founded Email(s): ssarah@gmail.com ayesha@yahoo.com
```

```
Select the options from below-

 1- Convert to Upper Case.
 2- Convert to Lower Case.
 3- Encrypt Your Text.
 4- Decrypt Your Text.
 5- Remove Extra Spacing.
 6- Remove all Numbers.
 0- Press 0 for Exit

Enter your choice :  1

Upper Case Text is : HELLO, MY NAME IS SARAH. MY EMAIL IS SSARAH@GMAIL.COM &      420    MY FRIENDS MAIL IS AYESHA@YAHOO.COM 6949.
  WE WORK48 AS A CONTENT-CREATOR         FROM PAST 2 YEARS.


Select the options from below-

 1- Convert to Upper Case.
 2- Convert to Lower Case.
 3- Encrypt Your Text.
 4- Decrypt Your Text.
 5- Remove Extra Spacing.
 6- Remove all Numbers.
 0- Press 0 for Exit
```

```
Enter your choice :  2

Lower Case Text is : hello, my name is sarah. my email is ssarah@gmail.com &       420     my friends mail is ayesha@yahoo.com 6949.
  we work48 as a content-creator          from past 2 years.


Select the options from below-

 1- Convert to Upper Case.
 2- Convert to Lower Case.
 3- Encrypt Your Text.
 4- Decrypt Your Text.
 5- Remove Extra Spacing.
 6- Remove all Numbers.
 0- Press 0 for Exit

Enter your choice :  3

       The Encrypted Text is : mjqqt1%r~%sfrj%nx%xfwfm3%r~%jrfnq%nx%xxfwfmElrfnq3htr%+%%%%%%975%%%%r~%kwnjsix%rfnq%nx%f~jxmfE~fmtt
9>3%%%%%%%%%|j%|twp9=%fx%f%htsyjsy2hwjfytw%%%%%%%%%kwtr%ufxy%7%~jfwx3
```

```
Select the options from below-

 1- Convert to Upper Case.
 2- Convert to Lower Case.
 3- Encrypt Your Text.
 4- Decrypt Your Text.
 5- Remove Extra Spacing.
 6- Remove all Numbers.
 0- Press 0 for Exit

Enter your choice :  3

       The Encrypted Text is : rovvy6*w?xkwo*s}*}k|kr8*w?owksv*s}*}}k|krJqwksv8myw*0******><:*****w?p|soxn}*wksv*s}*k?}rkJ?kryy8
>C8*********?*?|u>B*k}*k*myx~ox~7m|ok~y|*********p|yw*zk}~*<*?k|}8


Select the options from below-

 1- Convert to Upper Case.
 2- Convert to Lower Case.
 3- Encrypt Your Text.
 4- Decrypt Your Text.
 5- Remove Extra Spacing.
 6- Remove all Numbers.
 0- Press 0 for Exit

Enter your choice :  4

       The Decrypted Text is : mjqqt1%r~%sfrj%nx%xfwfm3%r~%jrfnq%nx%xxfwfmE1rfnq3htr%+%%%%%%975%%%%r~%kwnjsix%rfnq%nx%f~jxmfE~fmtt3
9>3%%%%%%%%%|j%|twp9=%fx%f%htsyjsy2hwjfytw%%%%%%%%%kwtr%ufxy%7%~jfwx3
```

```
Enter your choice :  4

       The Decrypted Text is : hello, my name is sarah. my email is ssarah@gmail.com &      420     my friends mail is ayesha@yahoo.com 69
49.         we work48 as a content-creator         from past 2 years.


Select the options from below-

 1- Convert to Upper Case.
 2- Convert to Lower Case.
 3- Encrypt Your Text.
 4- Decrypt Your Text.
 5- Remove Extra Spacing.
 6- Remove all Numbers.
 0- Press 0 for Exit

Enter your choice :  5

Without Extra Spaces Text : hello, my name is sarah. my email is ssarah@gmail.com & 420 my friends mail is ayesha@yahoo.com 6949. we work48
as a content-creator from past 2 years.
```

```
Select the options from below-

 1- Convert to Upper Case.
 2- Convert to Lower Case.
 3- Encrypt Your Text.
 4- Decrypt Your Text.
 5- Remove Extra Spacing.
 6- Remove all Numbers.
 0- Press 0 for Exit

Enter your choice :  6

Without Numbers Text : hello, my name is sarah. my email is ssarah@gmail.com & 0 my friends mail is ayesha@yahoo.com . we work as a content-
creator from past  years.
```

```
Enter your choice :  1

Upper Case Text is : HELLO, MY NAME IS SARAH. MY EMAIL IS SSARAH@GMAIL.COM & 0 MY FRIENDS MAIL IS AYESHA@YAHOO.COM . WE WORK AS A CONTENT-
EATOR FROM PAST  YEARS.


Select the options from below-

 1- Convert to Upper Case.
 2- Convert to Lower Case.
 3- Encrypt Your Text.
 4- Decrypt Your Text.
 5- Remove Extra Spacing.
 6- Remove all Numbers.
 0- Press 0 for Exit

Enter your choice :  0

       PROGRAM SUCCESSFULLY TERMINATED!
```

Answers To The Given Rubrics:

i. PROBLEM IDENTIFICATION:

Scenario: Sarah is a content writer working for a digital marketing agency. She often receives large amounts of text from different sources, such as client submissions, team collaborations, and online research. However, Sarah faces several challenges while working with this text, including inconsistencies, formatting issues, and the need to extract specific information. To overcome these challenges, Sarah decides to develop a text analyzer application.

Problem: Sarah's problem is that she spends a significant amount of time manually cleaning and processing text, which hampers her productivity and efficiency. Inconsistent formatting, unnecessary numbers, and extraneous spacing make it difficult for her to work with the text effectively. Additionally, she often needs to identify email addresses within the text, which is a time-consuming task.

DESIGNATION OF Text Analyzer Application To Solve Problem:

The text analyzer application is designed to address these challenges by offering features such as text cleaning, formatting assistance, email extraction, and encryption. It aims to streamline the text processing workflow, save time, and enhance the accuracy and reliability of the data analysis process. By utilizing this application, Sarah can overcome the limitations posed by unstructured text, improve data quality, and derive valuable insights more efficiently, ultimately benefiting her work as a data analyst and the market research company she works for.

ii. OOPS REQUIREMENT IDENTIFICATION:

In our group project, we have effectively applied various object-oriented programming (OOP) principles to solve the problem. Let's discuss the extent to which these principles have been utilized:

1) **Abstraction**: As a team, we have employed abstraction by defining abstract base classes, including **EncryptDecryptBase**, **TextModifierBase**, **LengthBase**, **WordsBase**, **SentencesBase**, and **Email**. These abstract classes provide a generalized structure and contain virtual

functions that need to be implemented by derived classes. This approach enhances code reusability and promotes modularity within our project.

```
1. // ------------Encrypt Decrypt Modifier Base Class-------
2. class EncryptDecryptBase
3. {
4. public:
5.     virtual void encrypt(string &str, int len) = 0;
6.     virtual void decrypt(string &str, int len) = 0;
7. };
8.
```

```
// ------------------------Text Modifier Base Class------------
class TextModifierBase
{
public:
    virtual void RemoveSpaces(string &str, int len) = 0;
    virtual void RemoveNumbers(string &str, int len) = 0;
    virtual void UpperCase(string &str, int len) = 0;
    virtual void LowerCase(string &str, int len) = 0;
};
```

2) **Encapsulation**: We have collectively encapsulated related data and functionality within classes. Each class represents a specific aspect of the problem domain, such as encryption/decryption, text modification, text analysis, and email handling. By encapsulating member functions and variables

within these classes, we ensure data integrity and promote code organization and maintainability.

```
// --------------EMAILS CLASS--------------
class Email
{
protected:
    string email;
    void setEmail(string email)
    {
        this->email = email;
    }
    string &getEmail() { return email; }
```

```
class SentencesBase
{
private:
    static int n;
};
int SentencesBase::n = 0;
```

3) **Inheritance**: Inheritance has been effectively utilized in our project. We have created derived classes, such as **EncryptDecrypt**, **TextModifier**, and **Count**, which inherit functionality from their respective base classes. This inheritance relationship establishes an "is-a" relationship, allowing us to reuse code from the base classes and build upon their functionality. This approach

promotes code reuse, enhances code structure, and simplifies maintenance.

```
class EncryptDecrypt : public EncryptDecryptBase…
```

```
class TextModifier : public TextModifierBase…
```

```
class Count : public WordsBase, public SentencesBase, public
LengthBase, public Email…
```

4) **Polymorphism**: We have made extensive use of polymorphism in our project. By defining virtual functions in the base classes and overriding them in the derived classes, such as **encrypt**, **decrypt**, **RemoveSpaces**, and **RemoveNumbers**, we achieve polymorphic behavior. This enables objects of different classes to exhibit different behaviors based on their specific implementations. This flexibility enhances extensibility and allows us to handle diverse text manipulation scenarios.

```
9.  // ----------NUMBER OF CHARACTERS---
10.         LengthBase() { n++; }
11.         virtual int count(string &str, int len)
12.         {
13.             int i = 0;
14.             for (i = 0; i < len; i++)
15.             {
16.             }
```

```
17.             return i;
18.         }
```

```
    // --------------NUMBER OF WORDS---------------
    virtual int count(string &str, int len)
    {
        int numberofwords = 01;
        string temp = "";
        for (int i = 0; i < len; i++)
        {
            if (str[i] == ' ')
            {
                if (temp != "")
                    numberofwords = numberofwords + 1;
                temp.clear();
            }
            else
            {
                temp += str[i];
            }
        }
        return numberofwords;
    }
     // --------------NUMBER OF SENTENCES---------------
    virtual int count(string &str, int len)
    {
        int numberofsentences = 0;
        for (int i = 0; i < len; i++)
        {
            if (str[i] == '.')
            {
                numberofsentences++;
            }
        }
        return numberofsentences;
```

```
    }
```

Overall, as a group, we have demonstrated a solid understanding of OOP principles in our project. Through abstraction, encapsulation, inheritance, and polymorphism, we have designed a solution that effectively addresses the problem requirements. Our collaborative effort has resulted in a well-structured and maintainable codebase that showcases the power of OOP in solving real-world problems.

iii. SOLUTION TO THE PROBLEM:

The solution to the problem is that Sarah develops a text analyzer application to streamline her text-processing tasks. The application offers a range of features that address her specific problems:

1. **Text cleaning**: The application allows Sarah to remove unnecessary numbers and extra spacing from the text, providing her with clean and consistent content to work with. This saves her time and ensures uniformity in the text across different sources.
2. **Formatting assistance**: Sarah can now convert the text to UPPERCASE or LOWERCASE as needed,

ensuring consistency to specific formatting requirements set by clients. The application automatically handles the formatting, relieving Sarah from manual efforts.

3. **Email extraction**: The text analyzer application includes a feature to identify and extract email addresses from the text. This functionality helps Sarah quickly locate and compile a list of email addresses without having to manually search through the text. It simplifies her task of organizing contacts or sending email communications.
4. **Encryption for data privacy**: As a content writer, Sarah often deals with sensitive information, such as client details or proprietary content. The application provides encryption and decryption capabilities to protect confidential information within the text. This ensures data privacy and security during communication or storage.

By utilizing the text analyzer application, Sarah can now significantly improve her productivity and efficiency. It automates time-consuming tasks, ensures consistent formatting, simplifies email extraction, and enhances data privacy.

INDIVIDUAL CONTRIBUTION:

Most of the work was done in the group but some individual contributions of the group members are:

MUHAMMAD ABDULLAH (CR-027):

- Main Idea and lead: Provide the central theme or problem that the project seeks to address and the solution of it.
- Project Planning and Syntax of Code.
- Main program logic and user interaction: Implemented the **main()** function, which handled user input, text processing, and the menu-based interaction with the user.
- Inclusion of necessary inheritance implementation.

SHEHERYAR AMIR (CR-008):

- User Interface Design: Design and implement an intuitive and user-friendly graphical user interface for the text analyzer program,
- Error Handling and Exception Management: Implement robust error handling and exception management mechanisms in the code
- Logics and Calculations.
- Design a UML Class Diagram of the Code.

SYED GHUFRAN RAZA (CR-028):

- Inclusion of Comments: Include comments to provide explanations and descriptions of the purpose of functions.
- Performance Optimization: Analyze and optimize the code for improved performance, including algorithmic improvements, memory management, and reducing computational complexity.
- Contribute In making the report of the project.

LIMITATIONS OF THE CODE:

i. **Limited Error Handling:** The code does not have comprehensive error handling mechanisms. It may not handle unexpected input or edge cases gracefully, leading to potential runtime errors or unexpected behavior.

ii. **Lack Of Input Validation:** The code assumes valid input from the user without explicitly validating it.

iii. **Incomplete Email Handling:** The found email approach may not cover all valid email address formats or handle cases where email addresses are embedded within text in a more complex manner.

iv. **Code Efficiency And Optimization:** Depending on the scale and performance requirements of project, it's worth considering potential code optimizations.

v. **Limited Scalability**: The current code may have limitations when scaling up to handle larger inputs or datasets.

THE END