"Assignment-1"

A REPORT SUBMITTED TO

THE NATIONAL INSTITUTE OF ENGINEERING, MYSURU

(An Autonomous Institute under VTU, Belagavi)

In partial fulfilment of the requirements for Cryptography (CS6C02), Sixth Semester

Bachelor of Engineering
in
Computer Science and Engineering

Submitted by
Mausam Borah (4NI20CS051)

To

Ramya S

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

THE NATIONAL INSTITUTE OF ENGINEERING

Mysore - 570008

2022-2023

Question 1. Implement Affine Cipher.

Code:

Code

main

Go to file t

- images
- 1_affine.cpp
- 2_exteuclidean.cpp
- 3_hashing.cpp
- 4_saltpassword.cpp
- 5_diffiehellman.cpp

Documentation · Share feedback

CryptographyAssignment / 1_affine.cpp

mausam30 Update 1_affine.cpp 516e256 · 47 minutes ago History

Code Blame 72 lines (60 loc) · 2.08 KB Raw

```cpp
#include <iostream>
#include <string>
#include <cctype>

std::string affine_encrypt(const std::string& plain_text, int a, int b) {
    std::string alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
    std::string encrypted_text = "";

    for (size_t i = 0; i < plain_text.length(); i++) {
        char ch = plain_text[i];
        if (std::isalpha(ch)) {
            char uppercase_ch = std::toupper(ch);
            int index = alphabet.find(uppercase_ch);
            int encrypted_index = (a * index + b) % 26;
            char encrypted_char = alphabet[encrypted_index];
            encrypted_text += encrypted_char;
        } else {
            encrypted_text += ch;
        }
    }

    return encrypted_text;
}

std::string affine_decrypt(const std::string& cipher_text, int a, int b) {
    std::string alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
    std::string decrypted_text = "";
    int a_inverse = 0;

    for (int i = 0; i < 26; i++) {
        if ((a * i) % 26 == 1) {
            a_inverse = i;
            break;
        }
    }
```

Code

main

Go to file t

- images
- 1_affine.cpp
- 2_exteuclidean.cpp
- 3_hashing.cpp
- 4_saltpassword.cpp
- 5_diffiehellman.cpp

Documentation · Share feedback

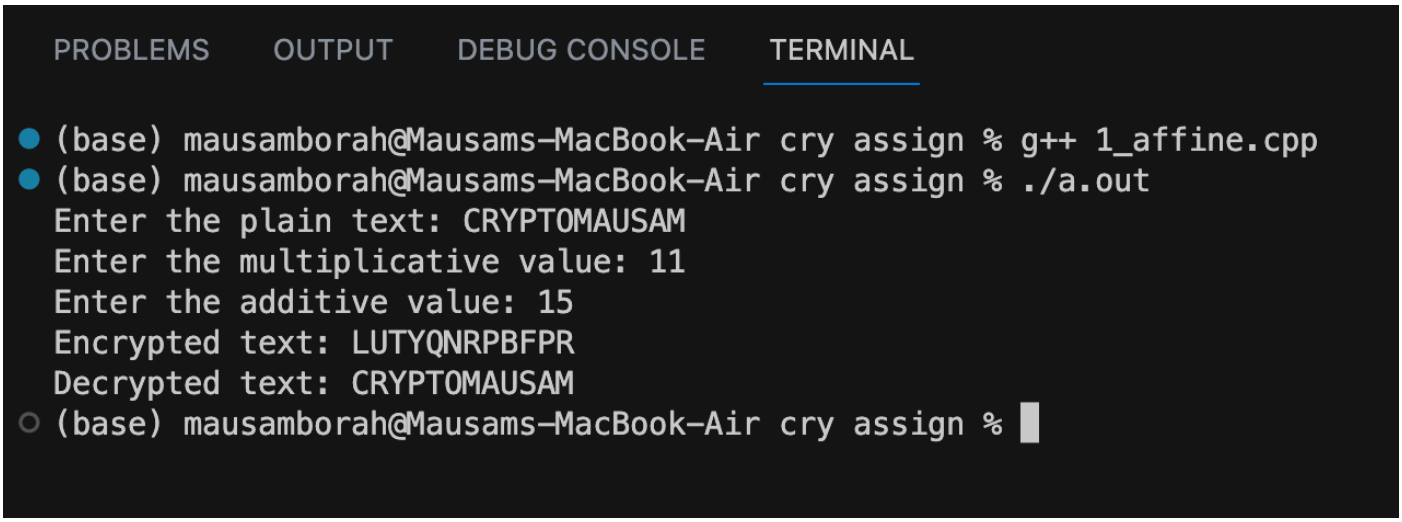
CryptographyAssignment / 1_affine.cpp ↑ Top

Code Blame 72 lines (60 loc) · 2.08 KB Raw

```cpp
    }

    for (size_t i = 0; i < cipher_text.length(); i++) {
        char ch = cipher_text[i];
        if (std::isalpha(ch)) {
            char uppercase_ch = std::toupper(ch);
            int index = alphabet.find(uppercase_ch);
            int decrypted_index = (a_inverse * (index - b + 26)) % 26;
            char decrypted_char = alphabet[decrypted_index];
            decrypted_text += decrypted_char;
        } else {
            decrypted_text += ch;
        }
    }

    return decrypted_text;
}

int main() {
    std::string plain_text;
    int a, b;

    std::cout << "Enter the plain text: ";
    std::getline(std::cin, plain_text);
    std::cout << "Enter the multiplicative value: ";
    std::cin >> a;
    std::cout << "Enter the additive value: ";
    std::cin >> b;
    std::cin.ignore();

    std::string encrypted_text = affine_encrypt(plain_text, a, b);
    std::cout << "Encrypted text: " << encrypted_text << std::endl;

    std::string decrypted_text = affine_decrypt(encrypted_text, a, b);
    std::cout << "Decrypted text: " << decrypted_text << std::endl;

    return 0;
}
```

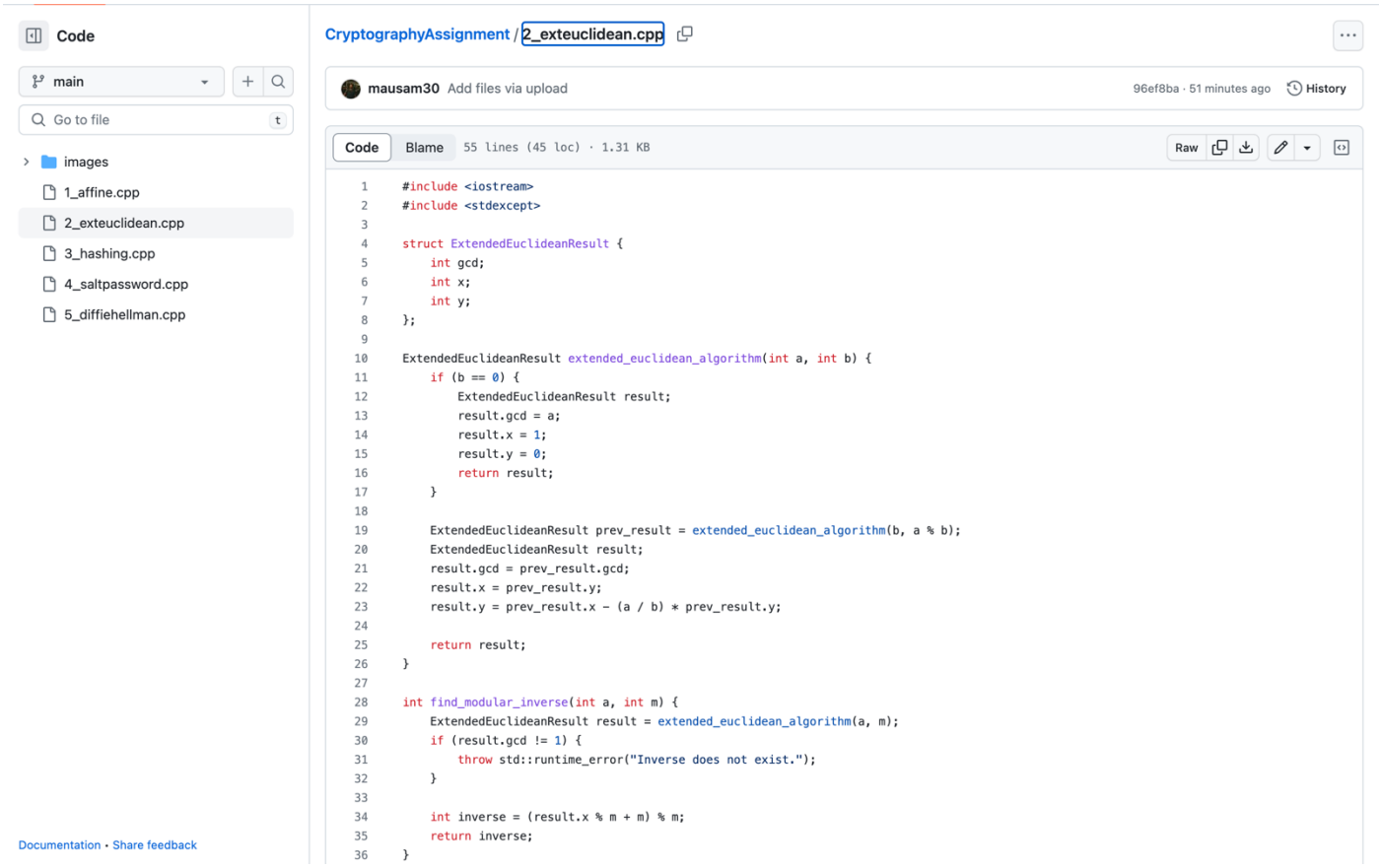
Documentation · Share feedback

Output:


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
(base) mausamborah@Mausams-MacBook-Air cry assign % g++ 1_affine.cpp
(base) mausamborah@Mausams-MacBook-Air cry assign % ./a.out
Enter the plain text: CRYPTOMAUSAM
Enter the multiplicative value: 11
Enter the additive value: 15
Encrypted text: LUTYQNRPBFPR
Decrypted text: CRYPTOMAUSAM
(base) mausamborah@Mausams-MacBook-Air cry assign %

Question 2. Implement Extended Euclidean Algorithm.

Code:



Code

main

Go to file t

images

1_affine.cpp

2_exteuclidean.cpp

3_hashing.cpp

4_saltpassword.cpp

5_diffiehellman.cpp

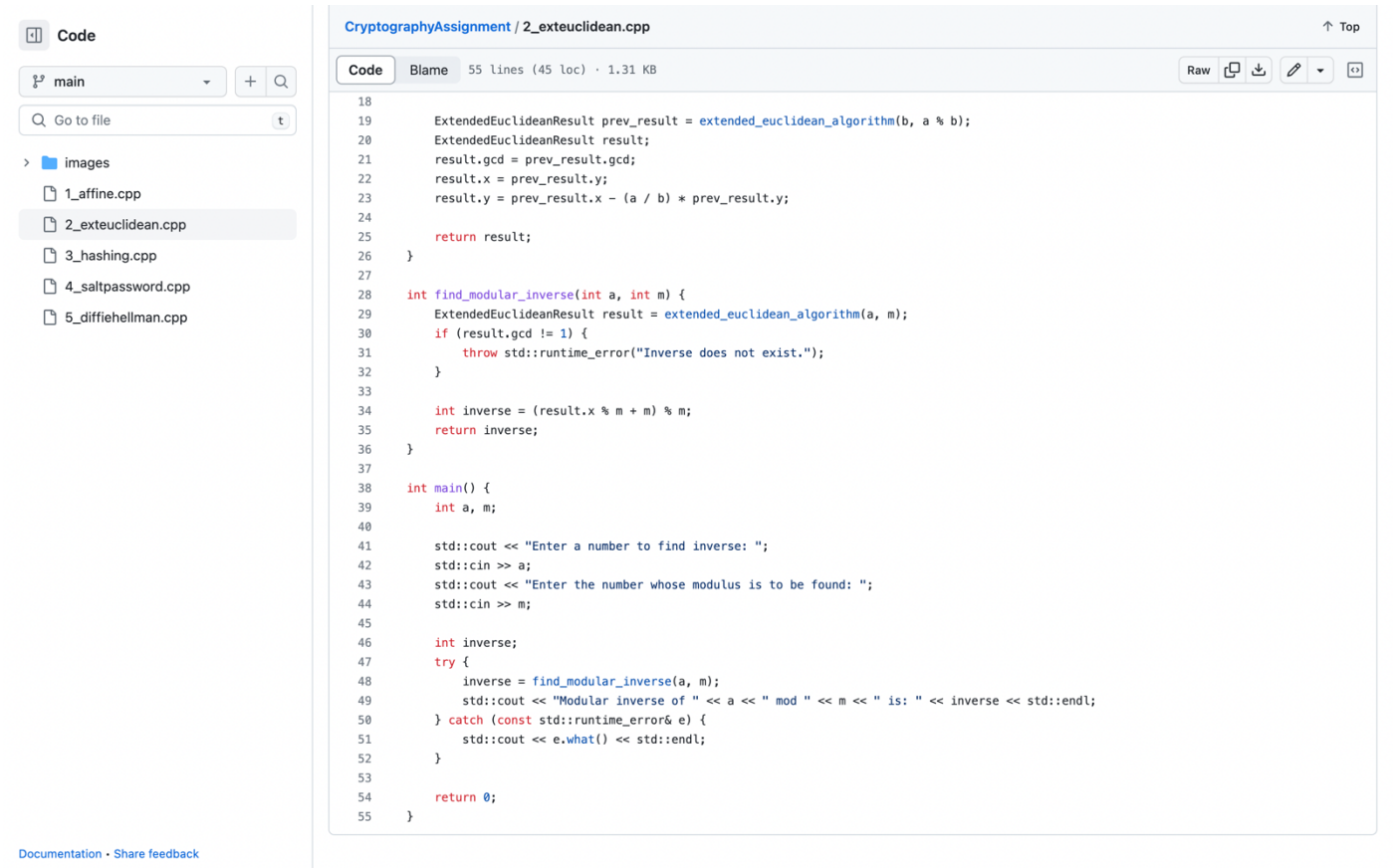
Documentation · Share feedback

CryptographyAssignment / 2_exteuclidean.cpp

mausam30 Add files via upload 96ef8ba · 51 minutes ago History

Code Blame 55 lines (45 loc) · 1.31 KB Raw

```cpp
#include <iostream>
#include <stdexcept>

struct ExtendedEuclideanResult {
    int gcd;
    int x;
    int y;
};

ExtendedEuclideanResult extended_euclidean_algorithm(int a, int b) {
    if (b == 0) {
        ExtendedEuclideanResult result;
        result.gcd = a;
        result.x = 1;
        result.y = 0;
        return result;
    }

    ExtendedEuclideanResult prev_result = extended_euclidean_algorithm(b, a % b);
    ExtendedEuclideanResult result;
    result.gcd = prev_result.gcd;
    result.x = prev_result.y;
    result.y = prev_result.x - (a / b) * prev_result.y;

    return result;
}

int find_modular_inverse(int a, int m) {
    ExtendedEuclideanResult result = extended_euclidean_algorithm(a, m);
    if (result.gcd != 1) {
        throw std::runtime_error("Inverse does not exist.");
    }

    int inverse = (result.x % m + m) % m;
    return inverse;
}
```



Code

main

Go to file t

images

1_affine.cpp

2_exteuclidean.cpp

3_hashing.cpp

4_saltpassword.cpp

5_diffiehellman.cpp

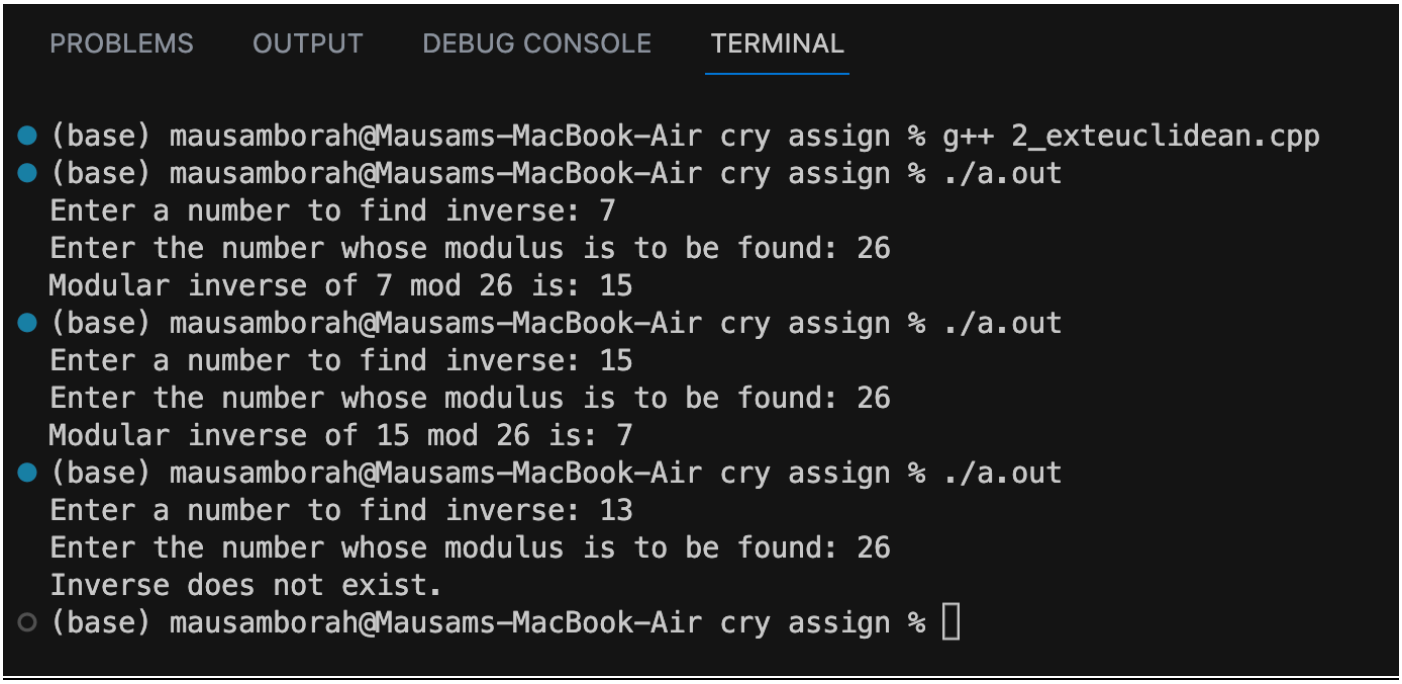
Documentation · Share feedback

CryptographyAssignment / 2_exteuclidean.cpp ↑ Top

Code Blame 55 lines (45 loc) · 1.31 KB Raw

```cpp
    ExtendedEuclideanResult prev_result = extended_euclidean_algorithm(b, a % b);
    ExtendedEuclideanResult result;
    result.gcd = prev_result.gcd;
    result.x = prev_result.y;
    result.y = prev_result.x - (a / b) * prev_result.y;

    return result;
}

int find_modular_inverse(int a, int m) {
    ExtendedEuclideanResult result = extended_euclidean_algorithm(a, m);
    if (result.gcd != 1) {
        throw std::runtime_error("Inverse does not exist.");
    }

    int inverse = (result.x % m + m) % m;
    return inverse;
}

int main() {
    int a, m;

    std::cout << "Enter a number to find inverse: ";
    std::cin >> a;
    std::cout << "Enter the number whose modulus is to be found: ";
    std::cin >> m;

    int inverse;
    try {
        inverse = find_modular_inverse(a, m);
        std::cout << "Modular inverse of " << a << " mod " << m << " is: " << inverse << std::endl;
    } catch (const std::runtime_error& e) {
        std::cout << e.what() << std::endl;
    }

    return 0;
}
```

Output:

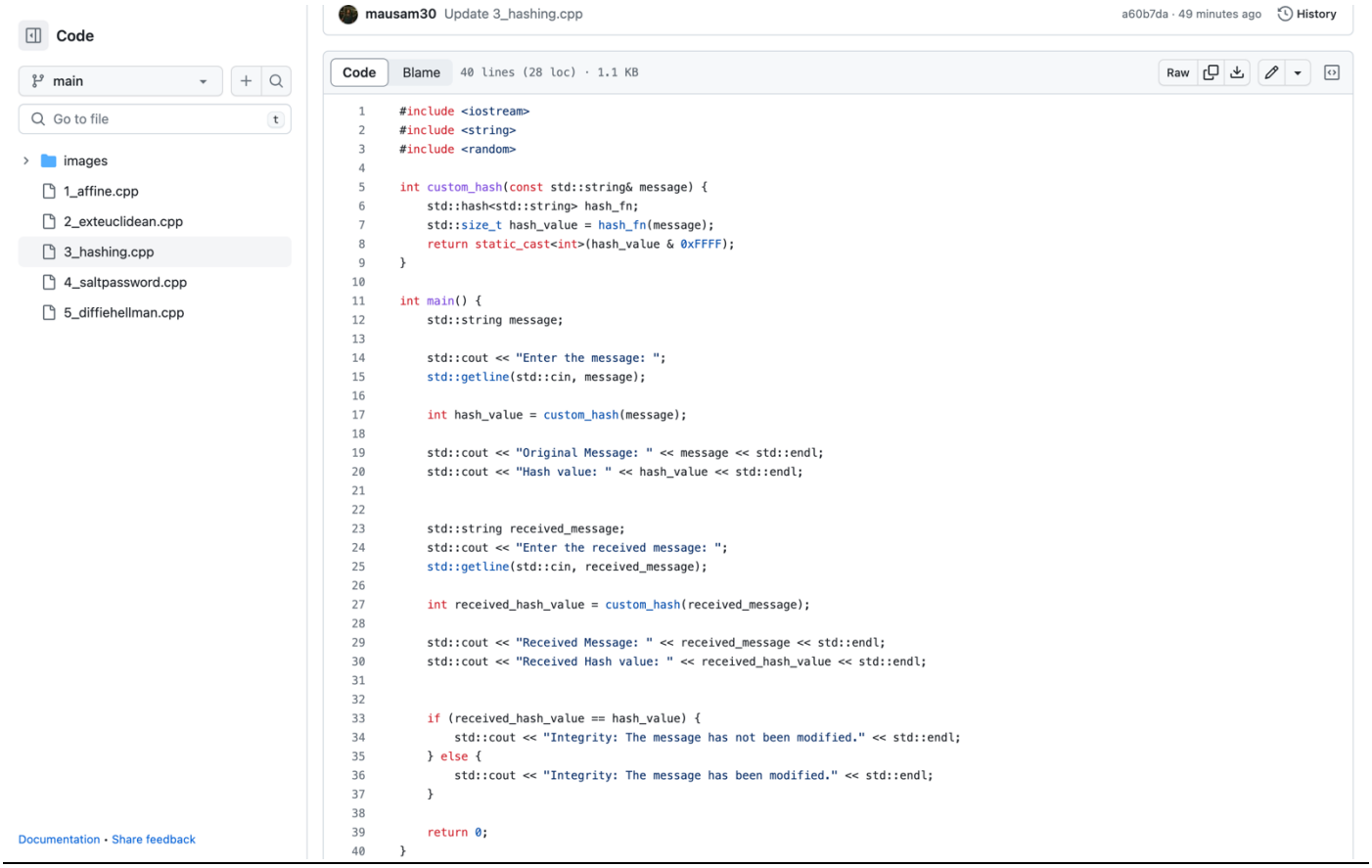

PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
(base) mausamborah@Mausams-MacBook-Air cry assign % g++ 2_exteuclidean.cpp
(base) mausamborah@Mausams-MacBook-Air cry assign % ./a.out
Enter a number to find inverse: 7
Enter the number whose modulus is to be found: 26
Modular inverse of 7 mod 26 is: 15
(base) mausamborah@Mausams-MacBook-Air cry assign % ./a.out
Enter a number to find inverse: 15
Enter the number whose modulus is to be found: 26
Modular inverse of 15 mod 26 is: 7
(base) mausamborah@Mausams-MacBook-Air cry assign % ./a.out
Enter a number to find inverse: 13
Enter the number whose modulus is to be found: 26
Inverse does not exist.
(base) mausamborah@Mausams-MacBook-Air cry assign %

Question 3. Select a simple message. Perform a hash function on it.

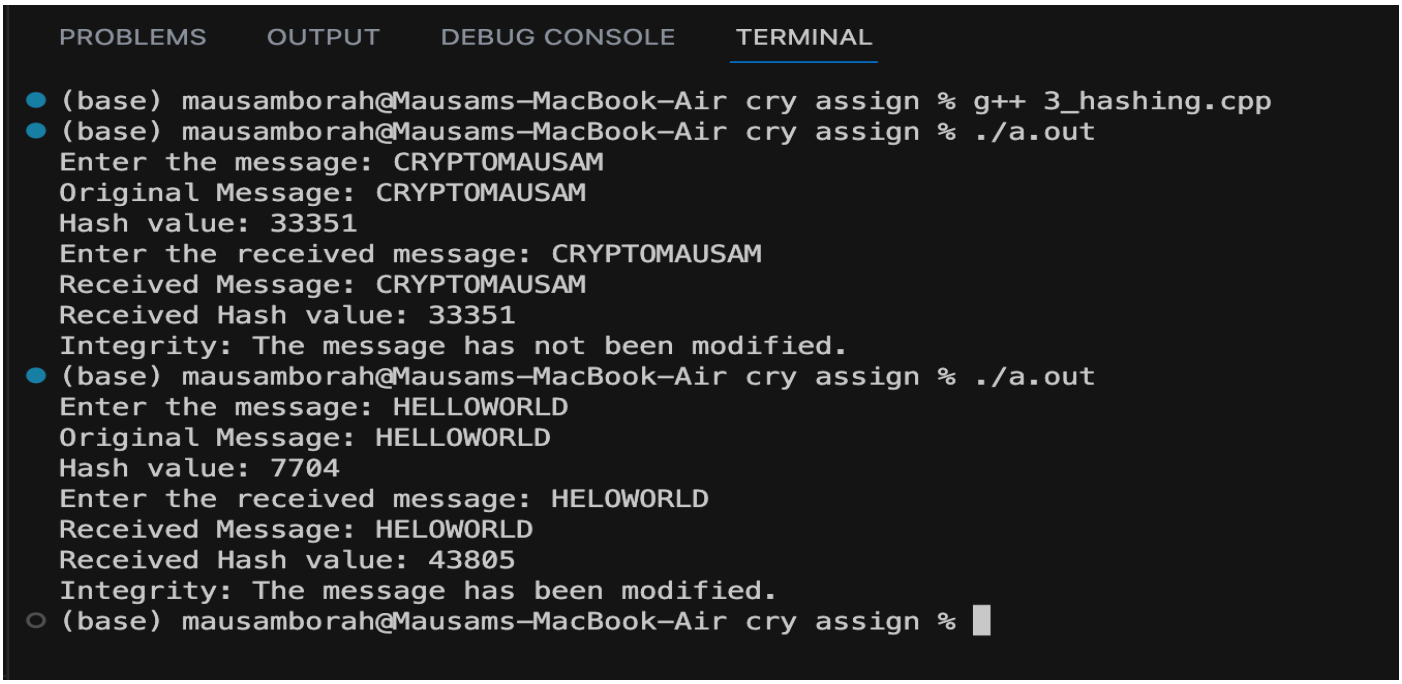
i) Simulate a receiver computing the hash again and ensuring its integrity.

ii) Slightly change the message. Simulate a receiver computing hash and find it not matching.

Code:



Output:

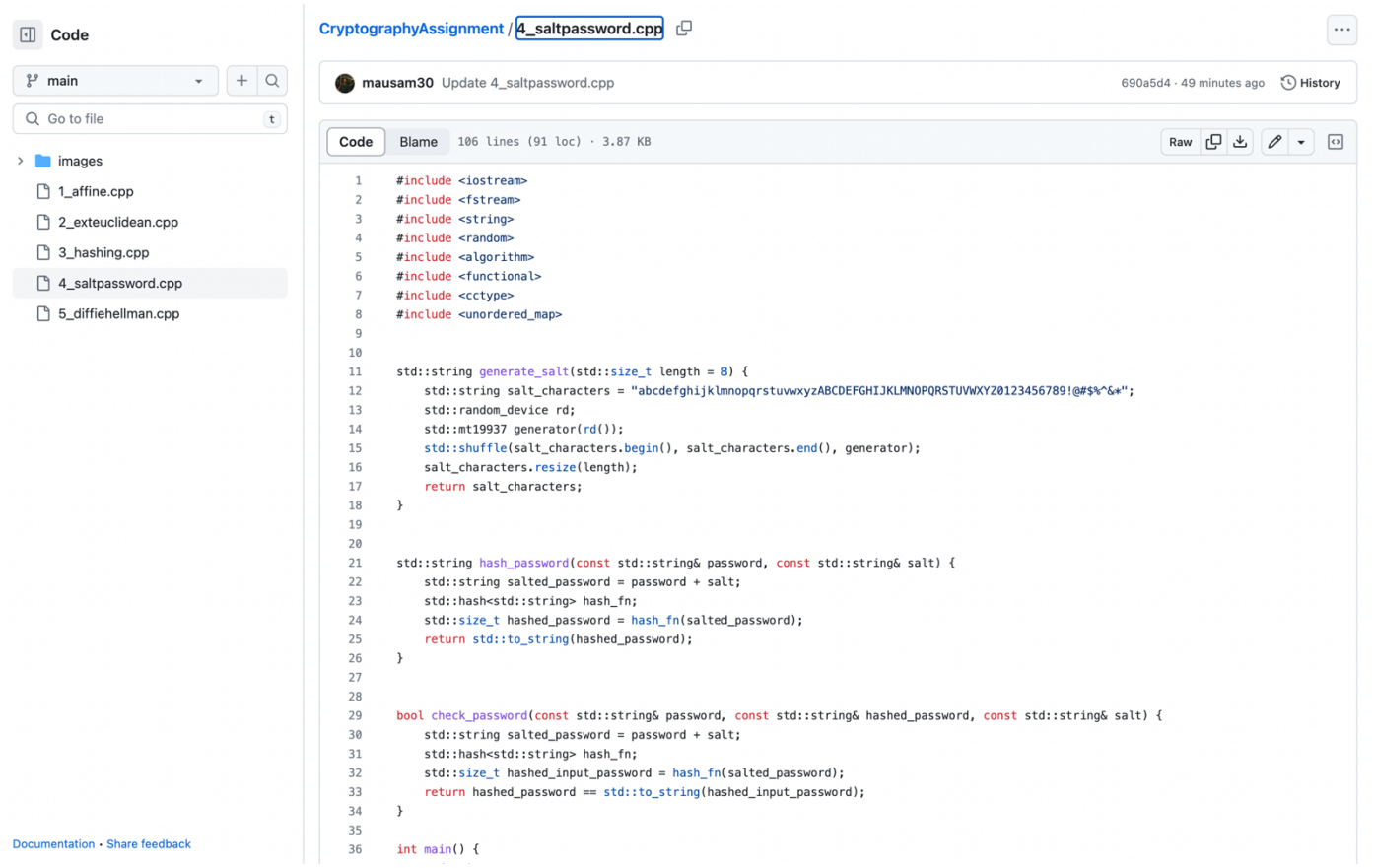


Question 4. a) Create a password file of 10 passwords and use it for identification.

b) Modify one to store the hash values of passwords & use it.

c) Optional: Create a salt file; add salt to password before storing in (b).

Code:



Code

main

Go to file t

- images
- 1_affine.cpp
- 2_exteuclidean.cpp
- 3_hashing.cpp
- 4_saltpassword.cpp
- 5_diffiehellman.cpp

Documentation · Share feedback

CryptographyAssignment / 4_saltpassword.cpp

mausam30 Update 4_saltpassword.cpp 690a5d4 · 49 minutes ago History

Code Blame 106 lines (91 loc) · 3.87 KB Raw

```cpp
#include <iostream>
#include <fstream>
#include <string>
#include <random>
#include <algorithm>
#include <functional>
#include <cctype>
#include <unordered_map>


std::string generate_salt(std::size_t length = 8) {
    std::string salt_characters = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789!@#$%^&*";
    std::random_device rd;
    std::mt19937 generator(rd());
    std::shuffle(salt_characters.begin(), salt_characters.end(), generator);
    salt_characters.resize(length);
    return salt_characters;
}


std::string hash_password(const std::string& password, const std::string& salt) {
    std::string salted_password = password + salt;
    std::hash<std::string> hash_fn;
    std::size_t hashed_password = hash_fn(salted_password);
    return std::to_string(hashed_password);
}


bool check_password(const std::string& password, const std::string& hashed_password, const std::string& salt) {
    std::string salted_password = password + salt;
    std::hash<std::string> hash_fn;
    std::size_t hashed_input_password = hash_fn(salted_password);
    return hashed_password == std::to_string(hashed_input_password);
}

int main() {
```

Code

main

Go to file t

- images
- 1_affine.cpp
- 2_exteuclidean.cpp
- 3_hashing.cpp
- 4_saltpassword.cpp
- 5_diffiehellman.cpp

Documentation • Share feedback

CryptographyAssignment / 4_saltpassword.cpp ↑ Top

Code Blame 106 lines (91 loc) · 3.87 KB Raw

```cpp
int main() {
    std::size_t num_users;
    std::cout << "Enter the number of users: ";
    std::cin >> num_users;
    std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');

    std::ofstream file("hashed_passwords.txt");
    if (!file) {
        std::cerr << "Failed to open the file." << std::endl;
        return 1;
    }

    std::unordered_map<std::string, std::string> password_file;
    for (std::size_t i = 1; i <= num_users; ++i) {
        std::string username, password;
        std::cout << "Enter the username for User" << i << ": ";
        std::getline(std::cin, username);
        std::cout << "Enter the password for User" << i << ": ";
        std::getline(std::cin, password);
        password_file[username] = password;
    }

    std::unordered_map<std::string, std::pair<std::string, std::string> > hashed_password_file;
    for (std::pair<const std::string, std::string>& pair : password_file) {
        const std::string& username = pair.first;
        std::string& password = pair.second;
        const std::string salt = generate_salt();
        const std::string hashed_password = hash_password(password, salt);
        hashed_password_file.insert(std::make_pair(username, std::make_pair(hashed_password, salt)));
        file << username << ':' << hashed_password << ':' << salt << '\n';
    }
    file.close();

    std::string input_username, input_password;
    std::cout << "Enter username: ";
    std::getline(std::cin, input_username);
    std::cout << "Enter password: ";
    std::getline(std::cin, input_password);

    std::ifstream hashed_passwords_file("hashed_passwords.txt");
```

Code

main

Go to file t

- images
- 1_affine.cpp
- 2_exteuclidean.cpp
- 3_hashing.cpp
- 4_saltpassword.cpp
- 5_diffiehellman.cpp

Documentation • Share feedback

CryptographyAssignment / 4_saltpassword.cpp ↑ Top

Code Blame 106 lines (91 loc) · 3.87 KB Raw

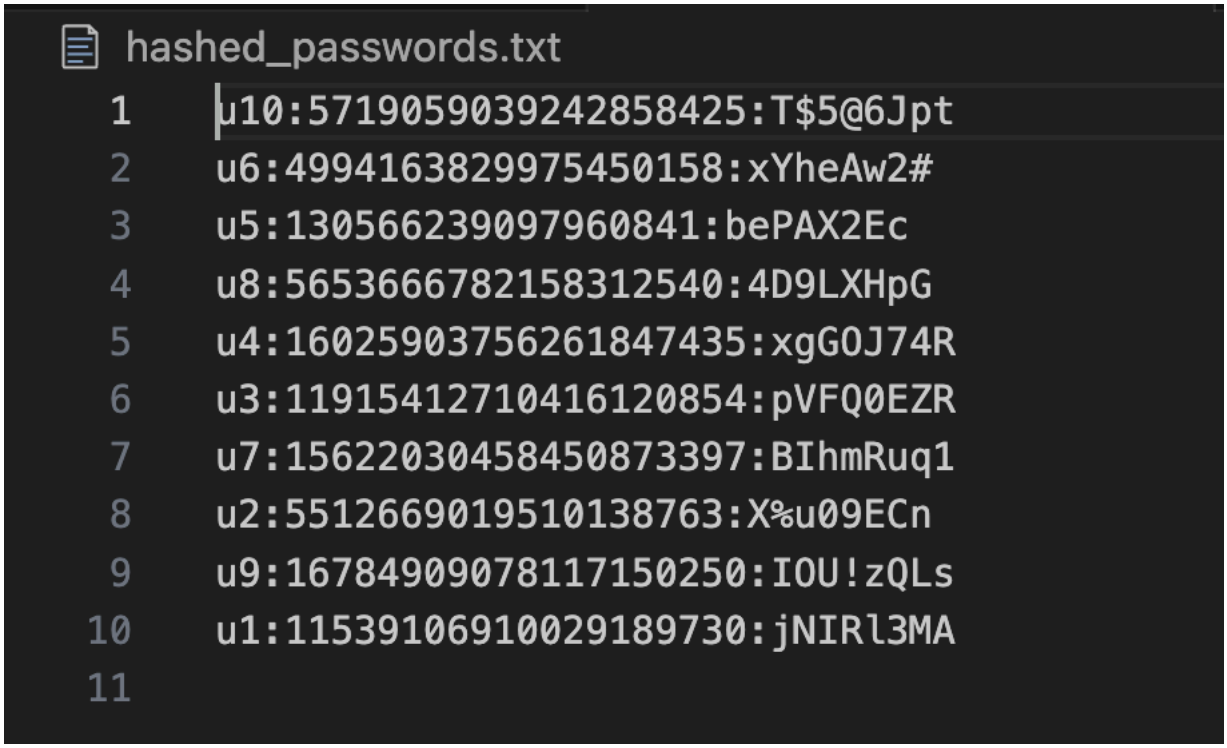
```cpp
    std::string input_username, input_password;
    std::cout << "Enter username: ";
    std::getline(std::cin, input_username);
    std::cout << "Enter password: ";
    std::getline(std::cin, input_password);

    std::ifstream hashed_passwords_file("hashed_passwords.txt");
    if (!hashed_passwords_file) {
        std::cerr << "Failed to open the file." << std::endl;
        return 1;
    }

    std::string line;
    bool login_successful = false;
    while (std::getline(hashed_passwords_file, line)) {
        std::size_t delimiter_pos_1 = line.find(':');
        std::size_t delimiter_pos_2 = line.find(':', delimiter_pos_1 + 1);
        if (delimiter_pos_1 != std::string::npos && delimiter_pos_2 != std::string::npos) {
            std::string stored_username = line.substr(0, delimiter_pos_1);
            std::string stored_hashed_password = line.substr(delimiter_pos_1 + 1, delimiter_pos_2 - delimiter_pos_1 - 1);
            std::string stored_salt = line.substr(delimiter_pos_2 + 1);
            if (stored_username == input_username &&
                check_password(input_password, stored_hashed_password, stored_salt)) {
                login_successful = true;
                break;
            }
        }
    }
    hashed_passwords_file.close();

    if (login_successful) {
        std::cout << "Login successful." << std::endl;
    } else {
        std::cout << "Login failed. Invalid username or password." << std::endl;
    }

    return 0;
}
```

Output:

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

(base) mausamborah@Mausams-MacBook-Air cry assign % ./a.out
Enter the number of users: 10
Enter the username for User1: u1
Enter the password for User1: u1pass
Enter the username for User2: u2
Enter the password for User2: u2pass
Enter the username for User3: u3
Enter the password for User3: u3pass
Enter the username for User4: u4
Enter the password for User4: u4pass
Enter the username for User5: u5
Enter the password for User5: u5pass
Enter the username for User6: u6
Enter the password for User6: u6pass
Enter the username for User7: u7
Enter the password for User7: u7pass
Enter the username for User8: u8
Enter the password for User8: u8pass
Enter the username for User9: u9
Enter the password for User9: u9pass
Enter the username for User10: u10
Enter the password for User10: u10pass
Enter username: u7
Enter password: u7pass
Login successful.
(base) mausamborah@Mausams-MacBook-Air cry assign %
```

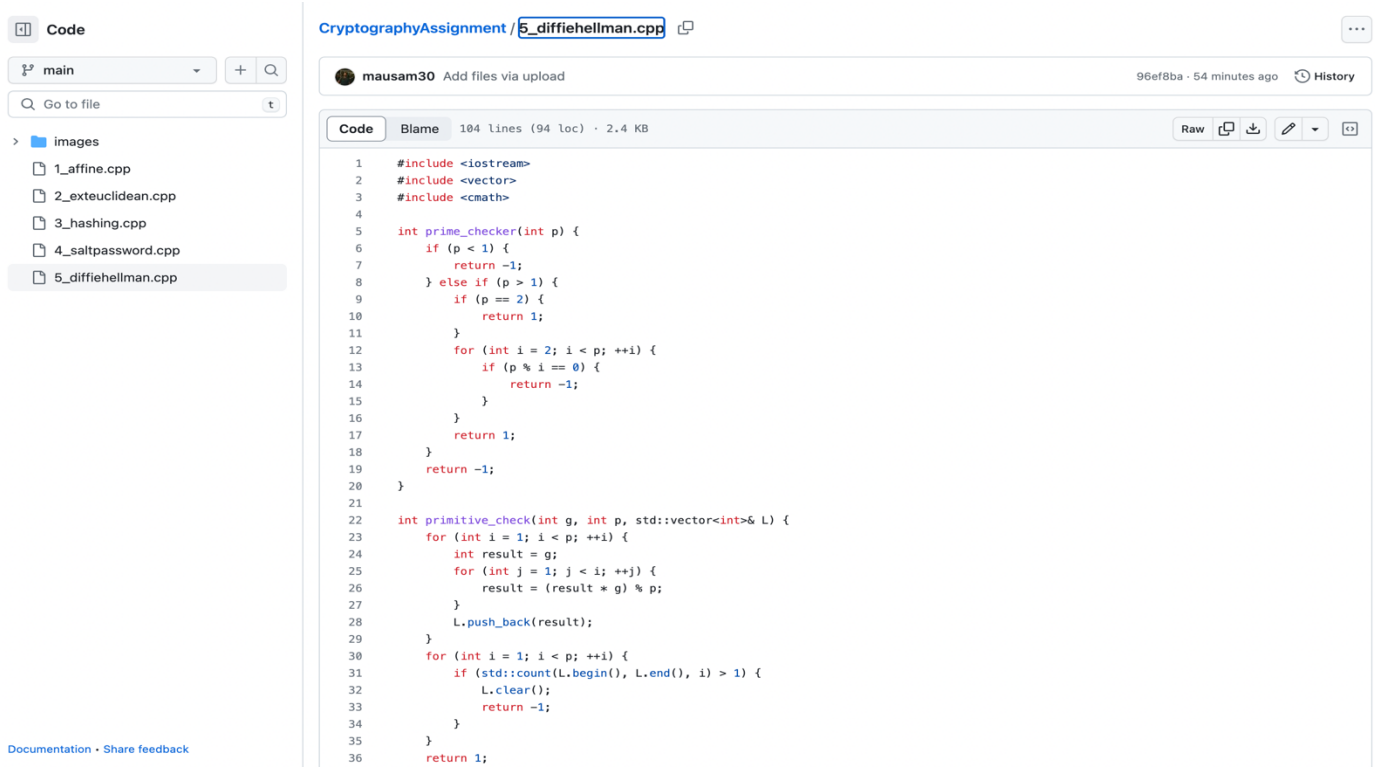
```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

(base) mausamborah@Mausams-MacBook-Air cry assign % ./a.out
Enter the number of users: 10
Enter the username for User1: u1
Enter the password for User1: u1pass
Enter the username for User2: u2
Enter the password for User2: u2pass
Enter the username for User3: u3
Enter the password for User3: u3pass
Enter the username for User4: u4
Enter the password for User4: u4pass
Enter the username for User5: u5
Enter the password for User5: u5pass
Enter the username for User6: u6
Enter the password for User6: u6pass
Enter the username for User7: u7
Enter the password for User7: u7pass
Enter the username for User8: u8
Enter the password for User8: u8pass
Enter the username for User9: u9
Enter the password for User9: u9pass
Enter the username for User10: u10
Enter the password for User10: u10pass
Enter username: u6
Enter password: u7pass
Login failed. Invalid username or password.
(base) mausamborah@Mausams-MacBook-Air cry assign %
```

Hashed password:



5. Generate a symmetric key by using Diffie-Hellman method given g, p, x and y

Code:



Code

main

Go to file t

- images
- 1_affine.cpp
- 2_exteuclidean.cpp
- 3_hashing.cpp
- 4_saltpassword.cpp
- 5_diffiehellman.cpp

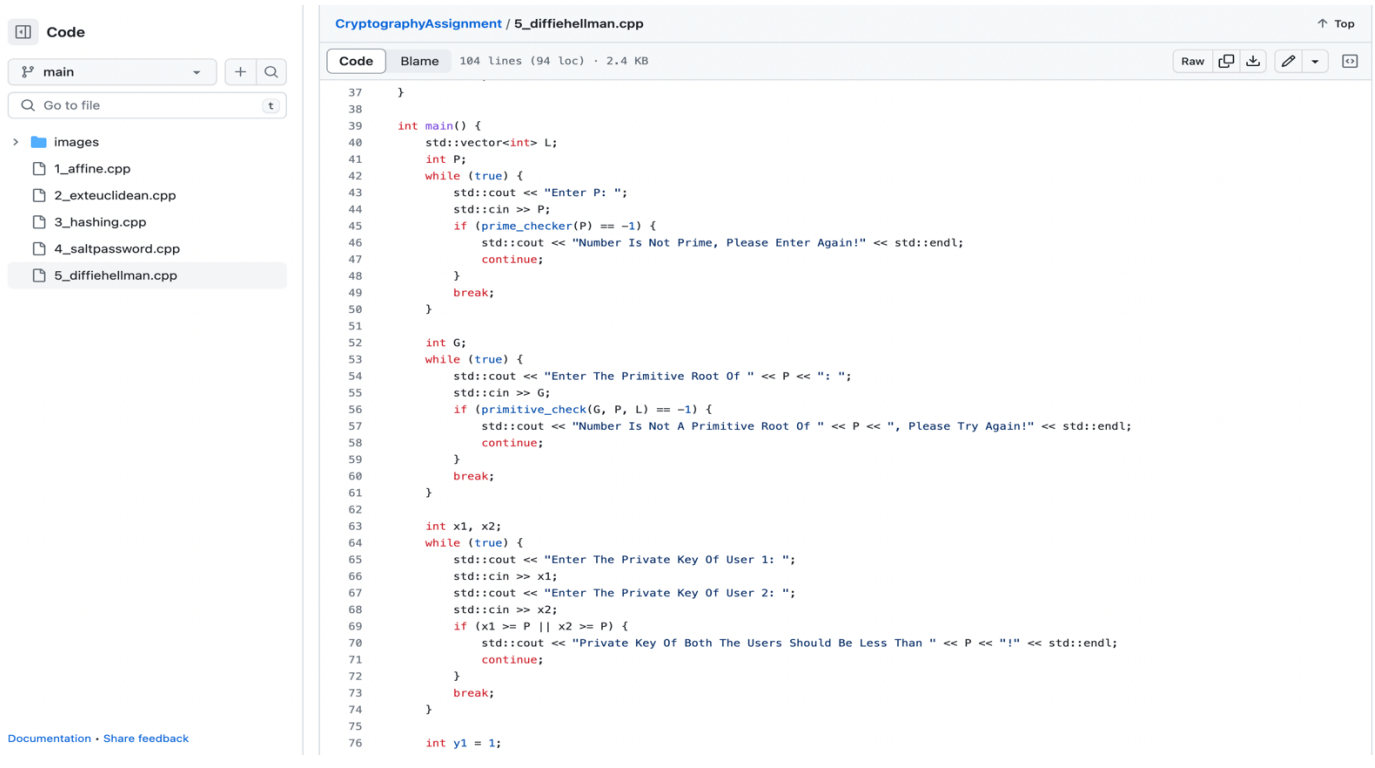
Documentation · Share feedback

CryptographyAssignment / 5_diffiehellman.cpp

mausam30 Add files via upload 96ef8ba · 54 minutes ago History

Code Blame 104 lines (94 loc) · 2.4 KB Raw

```cpp
#include <iostream>
#include <vector>
#include <cmath>

int prime_checker(int p) {
    if (p < 1) {
        return -1;
    } else if (p > 1) {
        if (p == 2) {
            return 1;
        }
        for (int i = 2; i < p; ++i) {
            if (p % i == 0) {
                return -1;
            }
        }
        return 1;
    }
    return -1;
}

int primitive_check(int g, int p, std::vector<int>& L) {
    for (int i = 1; i < p; ++i) {
        int result = g;
        for (int j = 1; j < i; ++j) {
            result = (result * g) % p;
        }
        L.push_back(result);
    }
    for (int i = 1; i < p; ++i) {
        if (std::count(L.begin(), L.end(), i) > 1) {
            L.clear();
            return -1;
        }
    }
    return 1;
```



Code

main

Go to file t

- images
- 1_affine.cpp
- 2_exteuclidean.cpp
- 3_hashing.cpp
- 4_saltpassword.cpp
- 5_diffiehellman.cpp

Documentation · Share feedback

CryptographyAssignment / 5_diffiehellman.cpp ↑ Top

Code Blame 104 lines (94 loc) · 2.4 KB Raw

```cpp
}

int main() {
    std::vector<int> L;
    int P;
    while (true) {
        std::cout << "Enter P: ";
        std::cin >> P;
        if (prime_checker(P) == -1) {
            std::cout << "Number Is Not Prime, Please Enter Again!" << std::endl;
            continue;
        }
        break;
    }

    int G;
    while (true) {
        std::cout << "Enter The Primitive Root Of " << P << ": ";
        std::cin >> G;
        if (primitive_check(G, P, L) == -1) {
            std::cout << "Number Is Not A Primitive Root Of " << P << ", Please Try Again!" << std::endl;
            continue;
        }
        break;
    }

    int x1, x2;
    while (true) {
        std::cout << "Enter The Private Key Of User 1: ";
        std::cin >> x1;
        std::cout << "Enter The Private Key Of User 2: ";
        std::cin >> x2;
        if (x1 >= P || x2 >= P) {
            std::cout << "Private Key Of Both The Users Should Be Less Than " << P << "!" << std::endl;
            continue;
        }
        break;
    }

    int y1 = 1;
```

Output:

```
PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL

(base) mausamborah@Mausams-MacBook-Air cry assign % g++ 5_diffiehellman.cpp
(base) mausamborah@Mausams-MacBook-Air cry assign % ./a.out
Enter P: 37
Enter The Primitive Root Of 37: 19
Enter The Private Key Of User 1: 5
Enter The Private Key Of User 2: 2

Secret Key For User 1 Is 3
Secret Key For User 2 Is 3
Keys Have Been Exchanged Successfully
(base) mausamborah@Mausams-MacBook-Air cry assign % ./a.out
Enter P: 47
Enter The Primitive Root Of 47: 23
Enter The Private Key Of User 1: 7
Enter The Private Key Of User 2: 4

Secret Key For User 1 Is 25
Secret Key For User 2 Is 25
Keys Have Been Exchanged Successfully
(base) mausamborah@Mausams-MacBook-Air cry assign % ./a.out
Enter P: 48
Number Is Not Prime, Please Enter Again!
Enter P: 23
Enter The Primitive Root Of 23: 17
Enter The Private Key Of User 1: 4
Enter The Private Key Of User 2: 3

Secret Key For User 1 Is 6
Secret Key For User 2 Is 6
Keys Have Been Exchanged Successfully
(base) mausamborah@Mausams-MacBook-Air cry assign %
```

Github Link : https://github.com/mausam30/CryptographyAssignment