**Charotar University of Science & Technology (CHARUSAT)**

**Devang Patel Institute of Advance Technology & Research (DEPSTAR)**

**Information Security (CE348)**

**Practical Solution**

|  |  |
| --- | --- |
| **Semester:6th** | **Academic Year: 2022-23** |

|  |  |
| --- | --- |
| **Sr No** | **Aim** |
| **1.**  **2.**  **3.**  **4.**  **5.**  **6.** | **The "Caesar Box," or "Caesar Cipher," is one of the earliest known ciphers. Developed around 100 BC, it was used by Julius Caesar to send secret messages to his generals in the field. In the event that one of his messages got intercepted, his opponent could not read them. This obviously gave him a great strategic advantage. Caesar shifted each letter of his message few letters to the right to produce what could be called the ciphertext. The ciphertext is what the enemy would see instead of the true message. So, for example, if Caesar’s messages were written in the English alphabet, and shift by 3 then each letter “A” in the message would become a "D," the "B’s" would become "E’s," and the "X's" become "A’s." This type of cipher is appropriately called a “shift cipher.” Implement the cipher in any programming language of your choice. Perform encryption, decryption. Discuss and try some possible attacks on traditional Caesar cipher.**  **Code:**  #include <iostream>  using namespace std;  string encrypt(string msg, int key);  string decrypt(string msg, int key);  void bruteForce(string cipherText);  int main()  {  string msg;  int key = 3;  cout << "Enter the Message: ";  cin >> msg;  string cipherText = encrypt(msg, key);  cout << endl  << "Cipher Text: " << cipherText << endl  << endl;  bruteForce(cipherText);  cout << "\n20DCE011 - Karan Bhatt";  }  string encrypt(string msg, int key)  {  string text;  for (int i = 0; i < msg.length(); i++)  text += 'a' + (msg[i] + key - 'a') % 26;  return text;  }  string decrypt(string msg, int key)  {  string text;  for (int i = 0; i < msg.length(); i++)  text += 'a' + (msg[i] - key - 'a' + 26) % 26;  return text;  }  void bruteForce(string cipherText)  {  for (int key = 1; key < 26; key++)  {  string text = decrypt(cipherText, key);  cout << "Key " << key << " : " << text << endl;  }  }  **Output screenshot :**    **The Playfair cipher was predominantly used by British forces during the Second Boer War (1899-1902) and World War I (1914-1918). Soldier from field wants to send message to base. Implement the cipher to encrypt and decrypt message.**  **Encrypt message: Hiroshima**  **Use key: pearlharbour**  **Code:**  #include <bits/stdc++.h>  using namespace std;  #define SIZE 30  // Function to convert the string to lowercase  void toLowerCase(char plain[], int ps)  {  int i;  for (i = 0; i < ps; i++) {  if (plain[i] > 64 && plain[i] < 91)  plain[i] += 32;  }  }  // Function to remove all spaces in a string  int removeSpaces(char\* plain, int ps)  {  int i, count = 0;  for (i = 0; i < ps; i++)  if (plain[i] != ' ')  plain[count++] = plain[i];  plain[count] = '\0';  return count;  }  // Function to generate the 5x5 key square  void generateKeyTable(char key[], int ks, char keyT[5][5])  {  int i, j, k, flag = 0;  // a 26 character hashmap  // to store count of the alphabet  int dicty[26] = { 0 };  for (i = 0; i < ks; i++) {  if (key[i] != 'j')  dicty[key[i] - 97] = 2;  }  dicty['j' - 97] = 1;  i = 0;  j = 0;  for (k = 0; k < ks; k++) {  if (dicty[key[k] - 97] == 2) {  dicty[key[k] - 97] -= 1;  keyT[i][j] = key[k];  j++;  if (j == 5) {  i++;  j = 0;  }  }  }  for (k = 0; k < 26; k++) {  if (dicty[k] == 0) {  keyT[i][j] = (char)(k + 97);  j++;  if (j == 5) {  i++;  j = 0;  }  }  }  }  void search(char keyT[5][5], char a, char b, int arr[])  {  int i, j;  if (a == 'j')  a = 'i';  else if (b == 'j')  b = 'i';  for (i = 0; i < 5; i++) {  for (j = 0; j < 5; j++) {  if (keyT[i][j] == a) {  arr[0] = i;  arr[1] = j;  }  else if (keyT[i][j] == b) {  arr[2] = i;  arr[3] = j;  }  }  }  }  // Function to find the modulus with 5  int mod5(int a) { return (a % 5); }  // Function to make the plain text length to be even  int prepare(char str[], int ptrs)  {  if (ptrs % 2 != 0) {  str[ptrs++] = 'z';  str[ptrs] = '\0';  }  return ptrs;  }  // Function for performing the encryption  void encrypt(char str[], char keyT[5][5], int ps)  {  int i, a[4];  for (i = 0; i < ps; i += 2) {  search(keyT, str[i], str[i + 1], a);  if (a[0] == a[2]) {  str[i] = keyT[a[0]][mod5(a[1] + 1)];  str[i + 1] = keyT[a[0]][mod5(a[3] + 1)];  }  else if (a[1] == a[3]) {  str[i] = keyT[mod5(a[0] + 1)][a[1]];  str[i + 1] = keyT[mod5(a[2] + 1)][a[1]];  }  else {  str[i] = keyT[a[0]][a[3]];  str[i + 1] = keyT[a[2]][a[1]];  }  }  }  // Function to encrypt using Playfair Cipher  void encryptByPlayfairCipher(char str[], char key[])  {  char ps, ks, keyT[5][5];  // Key  ks = strlen(key);  ks = removeSpaces(key, ks);  toLowerCase(key, ks);  // Plaintext  ps = strlen(str);  toLowerCase(str, ps);  ps = removeSpaces(str, ps);  ps = prepare(str, ps);  generateKeyTable(key, ks, keyT);  encrypt(str, keyT, ps);  }  int main()  {  char str[SIZE], key[SIZE];  // Key to be encrypted  strcpy(key, "pearlharbour");  cout << "Key text: " << key << "\n";  // Plaintext to be encrypted  strcpy(str, "Hiroshima");  cout << "Plain text: " << str << "\n";  // encrypt using Playfair Cipher  encryptByPlayfairCipher(str, key);  cout << "Cipher text: " << str << "\n";  cout << "\n20DCE011 - Karan Bhatt";  return 0;  }  **Output Screenshot:**    **The Rail Fence Cipher was invented in ancient times. It was used by the Greeks, who created a special tool, called scytale, to make message encryption and decryption easier. The letters are arranged in a way which is similar to the shape of the top edge of the rail fence. If king Leonidas want to sent message to Sparta as “300 achieved glory at hot gate, unite for Greece ” then what will be ciphertext when it is encrypted using 3 rows. Also implement decryption of message.**  **Code :**  #include<stdio.h>  #include<string.h>  void encryptMsg(char msg[], int key){  int msgLen = strlen(msg), i, j, k = -1, row = 0, col = 0;  char railMatrix[key][msgLen];  for(i = 0; i < key; ++i)  for(j = 0; j < msgLen; ++j) railMatrix[i][j] = '\n';  for(i = 0; i < msgLen; ++i){ railMatrix[row][col++] = msg[i];  if(row == 0 || row == key-1) k= k \* (-1);  row = row + k;  }  printf("\nEncrypted Message\n"); for(i = 0; i < key; ++i)  for(j = 0; j < msgLen; ++j)  if(railMatrix[i][j] != '\n')  printf("%c", railMatrix[i][j]);  }  void decryptMsg(char enMsg[], int key){  int msgLen = strlen(enMsg), i, j, k = -1, row = 0, col = 0, m = 0; char railMatrix[key][msgLen];  for(i = 0; i < key; ++i)  for(j = 0; j < msgLen; ++j) railMatrix[i][j] = '\n';  for(i = 0; i < msgLen; ++i){  railMatrix[row][col++] = '\*'; if(row == 0 || row == key-1)  k= k \* (-1);  row = row + k;  }  for(i = 0; i < key; ++i)  for(j = 0; j < msgLen; ++j) if(railMatrix[i][j] == '\*')  railMatrix[i][j] = enMsg[m++];  row = col = 0;  k = -1;  printf("\n\nDecrypted Message\n"); for(i = 0; i < msgLen; ++i){  printf("%c", railMatrix[row][col++]);  if(row == 0 || row == key-1)  k= k \* (-1);  row = row + k;  }  printf("\n20DCE011 - Karan Bhatt");  }  int main(){  char msg[] = "300 achieved glory at hot gate, unite for Greece"; char enMsg[] = "3ae rtttuere0 civdgoya o ae nt o ree0hel hg,ifGc"; int key = 3;  printf("\nOriginal Message\n%s \n", msg);  encryptMsg(msg, key);  decryptMsg(enMsg, key);  return 0;  }  **Output Screenshot:**    **Sergio wants to pass encrypted message to Rafael. He is using Hill cipher.**  **Message : family**  **Key : consider 3x3 matrix**  **Implement encryption and decryption of message.**  **Code:**  #include<iostream>  #include<math.h>  using namespace std;  float en[3][1], de[3][1], a[3][3], b[3][3], msg[3][1], m[3][3];  void getKeyMatrix()  {  int i, j;  char mes[3];  cout<<"Enter 3x3 matrix for key:\n";  for(i = 0; i < 3; i++)  for(j = 0; j < 3; j++) {  cin>>a[i][j];  m[i][j] = a[i][j];  }  cout<<"\nEnter a string: ";  cin>>mes;  for(i = 0; i < 3; i++)  msg[i][0] = mes[i] - 65;  }  //encrypts the message  void encrypt()  {  int i, j, k;  for(i = 0; i < 3; i++)  for(j = 0; j < 1; j++)  for(k = 0; k < 3; k++)  en[i][j] = en[i][j] + a[i][k] \* msg[k][j];  cout<<"\nEncrypted string:";  for(i = 0; i < 3; i++)  cout<<(char)(fmod(en[i][0], 26) + 65);  cout<<"\n";  }  //find inverse of key matrix  void inversematrix()  {  int i, j, k;  float p, q;  for(i = 0; i < 3; i++)  for(j = 0; j < 3; j++) {  if(i == j)  b[i][j]=1;  else  b[i][j]=0;  }  for(k = 0; k < 3; k++) {  for(i = 0; i < 3; i++) {  p = m[i][k];  q = m[k][k];  for(j = 0; j < 3; j++) {  if(i != k) {  m[i][j] = m[i][j]\*q - p\*m[k][j];  b[i][j] = b[i][j]\*q - p\*b[k][j];  }  }  }  }  for(i = 0; i < 3; i++)  for(j = 0; j < 3; j++)  b[i][j] = b[i][j] / m[i][i];  cout<<"\n\nInverse of the given key Matrix is:\n";  for(i = 0; i < 3; i++) {  for(j = 0; j < 3; j++)  cout<<b[i][j]<<" ";  cout<<"\n";  }  }  //decrypt the message  void decrypt()  {  int i, j, k;  inversematrix();  for(i = 0; i < 3; i++)  for(j = 0; j < 1; j++)  for(k = 0; k < 3; k++)  de[i][j] = de[i][j] + b[i][k] \* en[k][j];  cout<<"\nDecrypted string:";  for(i = 0; i < 3; i++)  cout<<(char)(fmod(de[i][0], 26) + 65);  cout<<"\n";  }  void line()  {  cout<<"-----------------------------------"<<endl;  }  int main()  {  string key;  cout<<"Enter the key:"<<endl;  cin>>key;  line();  getKeyMatrix();  line();  encrypt();  line();  decrypt();  line();  cout << "\n20DCE011 - Karan Bhatt";  }  **Output Screenshot:**    **Mr. Lucious Fox wants to transfer small amount of data within one session to Bruce**  **wayne. But they know that joker is listening/tapping to communication so they want**  **communication to be encrypted with secret key. Implement Diffie hellman algorithm to help them establishing key for session.**  **Code:**  #include <cmath>  #include <iostream>  using namespace std;  // Power function to return value of a ^ b mod P  long long int power(long long int a, long long int b,  long long int P)  {  if (b == 1)  return a;  else  return (((long long int)pow(a, b)) % P);  }  // Driver program  int main()  {  long long int P, G, x, a, y, b, ka, kb;    P = 11; // A prime number P is taken  cout << "The value of P : " << P << endl;  G = 7; // A primitive root for P, G is taken  cout << "The value of G : " << G << endl;    a = 6; // a is the chosen private key  cout << "The private key a for Lucious Fox is : " << a << endl;  x = power(G, a, P); // gets the generated key  b = 2;  cout << "The private key b for Bruce Wayne is : " << b << endl;  y = power(G, b, P); // gets the generated key  ka = power(y, a, P);  kb = power(x, b, P);  cout << "Secret key for the Lucious Fox is : " << ka << endl;  cout << "Secret key for the Bruce Wayne is : " << kb << endl;  cout << "\n20DCE011 - Karan Bhatt";  return 0;  }  **Output Screenshot:**    **After establishing connection with Bruce Wayne, established shared secret is used as an input to a random number generator available at both ends. Generated random numbers will follow same sequence at both ends. They are used as a one-time pad for encrypting/decrypting message. Message is converted to binary numbers and then encrypted with ex-or operation. Implement above system as a stream of message. Consider A=1, B=2, C=0…. So on.**  **Code:**  #include <iostream>  #include <string>  using namespace std;  int main() {  int key = 3;  string text, enc\_string = "", dec\_string = "";  char c;    cout << "Enter Text: ";  getline(cin, text);  cout << "Given Text: " << text << endl;  // Encryption  for (int i = 0; i < text.length(); i++) {  c = text[i];  enc\_string += char((int(c) + key - 97) % 26 + 97);  }  cout << "Encrypted String: " << enc\_string << endl;  // Decryption  for (int i = 0; i < enc\_string.length(); i++) {  c = enc\_string[i];  dec\_string += char((int(c) - key - 97) % 26 + 97);  }  cout << "Decrypted String: " << dec\_string << endl;  cout << "\n20DCE011 - Karan Bhatt";  return 0;  }  **Output Screenshot:** |