**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 <stdio.h>  int main() {  int key;  char msg[50],abc[50];  printf("enter message: ");  gets(msg);  printf("enter key between 0 to 26: ");  scanf("%d",&key);  printf("This is Encrypted Message of Entered String : ");  for(int i=0;msg[i]!='\0';i++)  {  msg[i]=msg[i]+key;  printf("%c",msg[i]);  abc[i]=msg[i];  }  printf("\n::::::::::::::::::::::::::::::::::::::");    for(int j=0;j<26;j++)  {  printf("\n Decrypted String No. %d : ",j);  for(int i=0;msg[i]!='\0';i++)  {  abc[i]=msg[i]-j;  printf("%c",abc[i]);  }  }  printf("\nCreated By : SHIVAM MEHTA-(20DCE060)");  return 0;  }    **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  void toLowerCase(char plain[], int ps)  {  int i;  for (i = 0; i < ps; i++) {  if (plain[i] > 64 && plain[i] < 91)  plain[i] += 32;  }  }  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;  }  void generateKeyTable(char key[], int ks, char keyT[5][5])  {  int i, j, k, flag = 0;  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;  }  }  }  }  int mod5(int a) { return (a % 5); }  int prepare(char str[], int ptrs)  {  if (ptrs % 2 != 0) {  str[ptrs++] = 'z';  str[ptrs] = '\0';  }  return ptrs;  }  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]];  }  }  }  void encryptByPlayfairCipher(char str[], char key[])  {  char ps, ks, keyT[5][5];  ks = strlen(key);  ks = removeSpaces(key, ks);  toLowerCase(key, ks);  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];  strcpy(key, "Hiroshima");  cout << "Key text: " << key << "\n";  strcpy(str, "pearlharbour");  cout << "Plain text: " << str << "\n";  encryptByPlayfairCipher(str, key);  cout << "Cipher text: " << str << "\n";  return 0;  }    **Output:**    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 <bits/stdc++.h>  using namespace std;  string encryptRailFence(string text, int key)  {  char rail[key][(text.length())];  for (int i=0; i < key; i++)  for (int j = 0; j < text.length(); j++)  rail[i][j] = '\n';  bool dir\_down = false;  int row = 0, col = 0;  for (int i=0; i < text.length(); i++)  {  if (row == 0 || row == key-1)  dir\_down = !dir\_down;  rail[row][col++] = text[i];  dir\_down?row++ : row--;  }  string result;  for (int i=0; i < key; i++)  for (int j=0; j < text.length(); j++)  if (rail[i][j]!='\n')  result.push\_back(rail[i][j]);  return result;  }  string decryptRailFence(string cipher, int key)  {  char rail[key][cipher.length()];  for (int i=0; i < key; i++)  for (int j=0; j < cipher.length(); j++)  rail[i][j] = '\n';  bool dir\_down;  int row = 0, col = 0;  for (int i=0; i < cipher.length(); i++)  {  if (row == 0)  dir\_down = true;  if (row == key-1)  dir\_down = false;  rail[row][col++] = '\*';  dir\_down?row++ : row--;  }  int index = 0;  for (int i=0; i<key; i++)  for (int j=0; j<cipher.length(); j++)  if (rail[i][j] == '\*' && index<cipher.length())  rail[i][j] = cipher[index++];  string result;  row = 0, col = 0;  for (int i=0; i< cipher.length(); i++)  {  if (row == 0)  dir\_down = true;  if (row == key-1)  dir\_down = false;  if (rail[row][col] != '\*')  result.push\_back(rail[row][col++]);  dir\_down?row++: row--;  }  return result;  }  int main()  {  cout << encryptRailFence("300 achieved glory at hot gate, unite for greece\n", 3) << endl;  cout << ":::::Decryption:::::"<<endl;  cout << decryptRailFence("3ae rtttuere0 civdgoya o ae nt o ree0hel hg,ifgc",3) << endl;  return 0;  }  **Output:**    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.  **Output:**  // C++ code to implement Hill Cipher  #include <iostream>  using namespace std;  void getKeyMatrix(string key, int keyMatrix[][3])  {  int k = 0;  for (int i = 0; i < 3; i++)  {  for (int j = 0; j < 3; j++)  {  keyMatrix[i][j] = (key[k]) % 65;  k++;  }  }  }  void encrypt(int cipherMatrix[][1],  int keyMatrix[][3],  int messageVector[][1])  {  int x, i, j;  for (i = 0; i < 3; i++)  {  for (j = 0; j < 1; j++)  {  cipherMatrix[i][j] = 0;    for (x = 0; x < 3; x++)  {  cipherMatrix[i][j] +=  keyMatrix[i][x] \* messageVector[x][j];  }    cipherMatrix[i][j] = cipherMatrix[i][j] % 26;  }  }  }  void HillCipher(string message, string key)  {  int keyMatrix[3][3];  getKeyMatrix(key, keyMatrix);  int messageVector[3][1];  for (int i = 0; i < 3; i++)  messageVector[i][0] = (message[i]) % 65;  int cipherMatrix[3][1];  encrypt(cipherMatrix, keyMatrix, messageVector);  string CipherText;  for (int i = 0; i < 3; i++)  CipherText += cipherMatrix[i][0] + 65;  cout << " Ciphertext:" << CipherText;  }  int main()  {  string message = "family";  string key = "GYBNQKURP";  HillCipher(message, key);  return 0;  }  **Output:**    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:**  import random  p = int(input('Enter a prime number : '))  g = int(input('Enter a number : '))  class A:  def \_\_init\_\_(self):  self.n = random.randint(1, p)  def publish(self):  return (g\*\*self.n)%p  def compute\_secret(self, gb):  return (gb\*\*self.n)%p  class B:  def \_\_init\_\_(self):  self.a = random.randint(1, p)  self.b = random.randint(1, p)  self.arr = [self.a,self.b]  def publish(self, i):  return (g\*\*self.arr[i])%p  def compute\_secret(self, ga, i):  return (ga\*\*self.arr[i])%p  lucious = A()  bruce = A()  eve = B()  print(f'Mr. Lucious Fox selected (a) : {lucious.n}')  print(f'Bruce wayne selected (b) : {bruce.n}')  print(f'Eve selected private number for Mr. Lucious Fox (c) : {eve.a}')  print(f'Eve selected private number for Bruce wayne (d) : {eve.b}')  ga = lucious.publish()  gb = bruce.publish()  gea = eve.publish(0)  geb = eve.publish(1)  print(f'Mr. Lucious Fox published (ga): {ga}')  print(f'Bruce wayne published (gb): {gb}')  print(f'Eve published value for Mr. Lucious Fox(gc): {gea}')  print(f'Eve published value for Bruce wayne (gd): {geb}')  sa = lucious.compute\_secret(gea)  sea = eve.compute\_secret(ga,0)  sb = bruce.compute\_secret(geb)  seb = eve.compute\_secret(gb,1)  print(f'Mr. Lucious Fox computed (S1) : {sa}')  print(f'Eve computed key for Mr. Lucious Fox (S1) : {sea}')  print(f'Bob computed (S2) : {sb}')  print(f'Eve computed key for Bruce wayne (S2) : {seb}')  **Output:**    After establishing connection with bruce wayne , established shared secret is used as a 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 one .  **Code:**  #include<stdio.h>  #include<math.h>  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);  }  int main()  {  long long int P, G, x, a, y, b, ka, kb;    P = 23;  printf("The value of P : %lld\n", P);  G = 9;  printf("The value of G : %lld\n\n", G);  a = 4;  printf("The private key a for Mr. Lucious Fox : %lld\n", a);  x = power(G, a, P);    b = 3;  printf("The private key b forBruce wayne : %lld\n\n", b);  y = power(G, b, P);  ka = power(y, a, P);  kb = power(x, b, P);  printf("Secret key for the Mr. Lucious Fox is : %lld\n", ka);  printf("Secret Key for the Bruce wayne is : %lld\n", kb);  return 0;  }  **Output:** |