**CSA5109-CRYPTOGRAPHY AND NETWORK SECURITY FOR DATA PROTECTION**

**LAB PROGRAMS**

**DATE:2-11-23**

**DAY: THURSDAY**

**1.Write a c program for Caesar cipher involves replacing each letter of the alphabet with the letter standing k places further down the alphabet,for k in the range 1 through 25.**

**PROGRAM:-**

**#include<stdio.h>**

**#include<string.h>**

**#include<ctype.h>**

**Int main(){**

**Int k=3,I,j;**

**Char input[1000],cipher[1000];**

**Printf(“Enter input string:”);**

**Scanf(“%s”,&input);**

**Printf(“Encrytption is : “);**

**For(i=0;i<strlen(input);i++){**

**If(islower(input[i])){**

**Cipher[i]=((input[i]-‘a’+k)%26)+’a’;**

**}**

**Else{**

**Cipher[i]=((input[i]-‘A’+k)%26)+’A’;**

**}**

**Printf(“%c”,cipher[i]);**

**}**

**Printf(“\nDecrytption is : “);**

**For( i=0;i<strlen(cipher);i++){**

**If(islower(input[i])){**

**Input[i]=((cipher[i]-‘a’-k)%26)+’a’;**

**}**

**Else{**

**Input[i]=((cipher[i]-‘A’-k)%26)+’A’;**

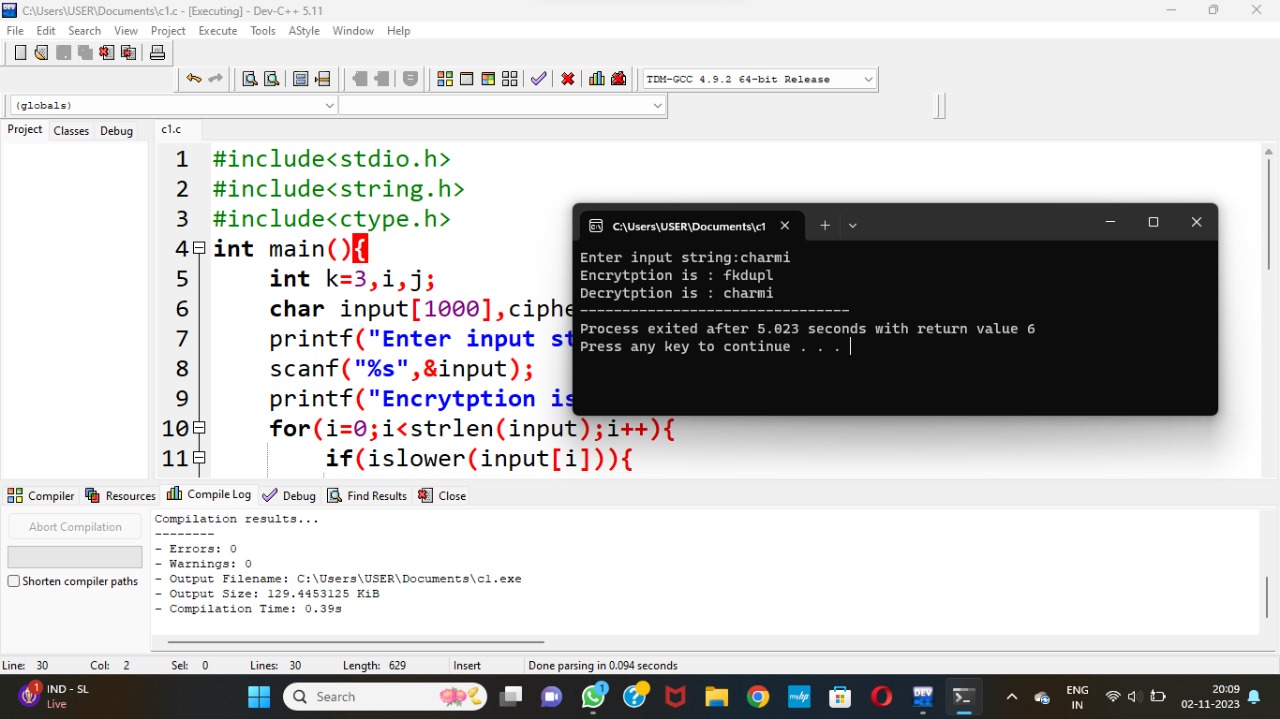
**}**

**Printf(“%c”,input[i]);**

**}**

**}**

**OUTPUT:-**

****

**2.Write a c program for monoalphabetic substitution cipher maps a plaintext alphabet to a cipher text alphabet , so that each letter of the plaintext alphabet maps to a single unique letter of the ciphertext alphabet.**

**PROGRAM:-**

**#include<stdio.h>**

**Int main(){**

**Int I,j;**

**Char alpha[100]=”abcdefghijklmnopqrstuvwxyz”,key[100]=”zyxwvutsrqponmlkjihgfedcba”,plain[100],cipher[100];**

**Int m=0,index[100];**

**Printf(“Enter plain text :”);**

**Scanf(“%s”,&plain);**

**For(i=0;i<strlen(plain);i++){**

**For(j=0;j<strlen(alpha);j++){**

**If(plain[i]==alpha[j]){**

**Index[m]=j;**

**M++;**

**}**

**}**

**}**

**Printf(“Cipher text: “);**

**For(i=0;i<strlen(plain);i++){**

**Cipher[i]=key[index[i]];**

**Printf(“%c”,cipher[i]);**

**}**

**Printf(“\n Plain text : “);**

**For(i=0;i<strlen(plain);i++){**

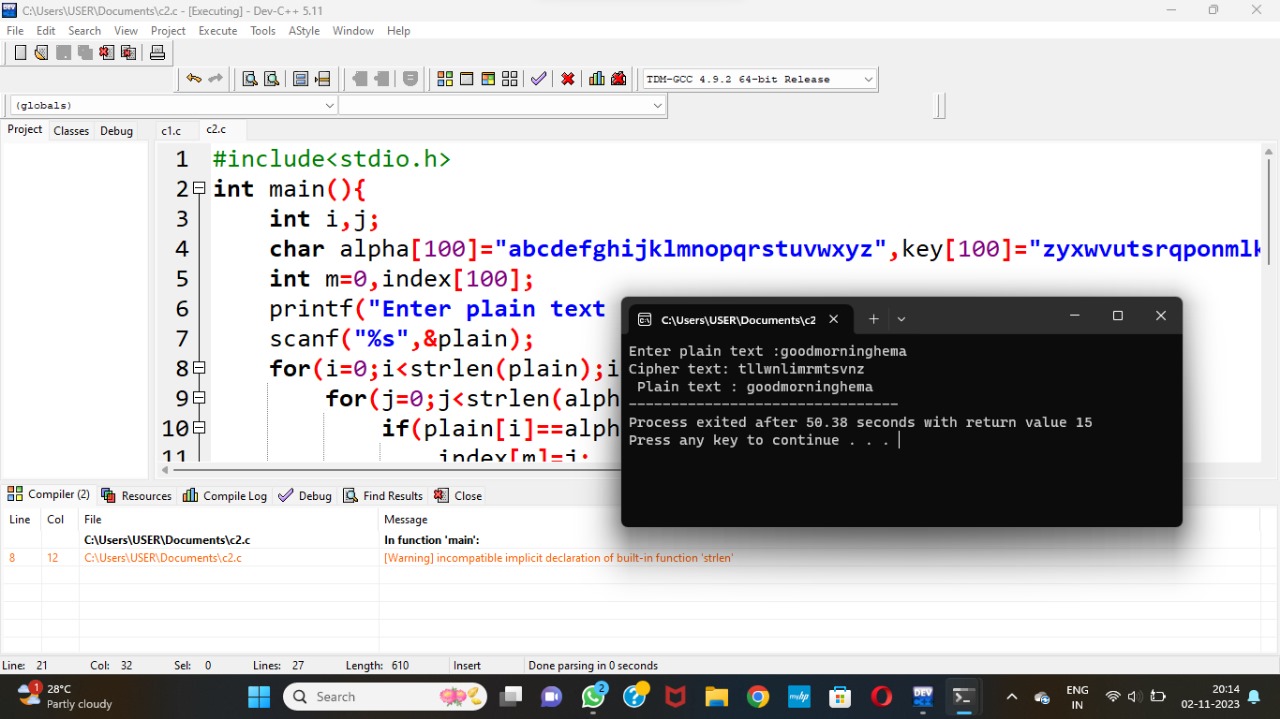
**Plain[i]=alpha[index[i]];**

**Printf(“%c”,plain[i]);**

**}**

**}**

**OUTPUT:-**

****

**3.Write a c program for playfair algorithm is based on the use of a 5x5 matrix of letters constructed using a keyword. Plaintext is encrypted two letters at a time using this matrix.**

**PROGRAM:-**

**#include<stdio.h>**

**#include<string.h>**

**#include<ctype.h>**

**int removerepeated(int size,int a[]);**

**int insertelementat(int position,int a[],int size);**

**main()**

**{**

**int i,j,k,numstr[100],numcipher[100],numkey[100],lenkey,templen,tempkey[100],flag=-1,size,cipherkey[5][5],lennumstr,row1,row2,col1,col2;**

**char str[100],key[100];**

**printf("Enter a string\n");**

**gets(str);**

**//converting entered string to Capital letters**

**for(i=0,j=0;i<strlen(str);i++)**

**{**

**if(str[i]!=' ')**

**{**

**str[j]=toupper(str[i]);**

**j++;**

**}**

**}**

**str[j]='\0';**

**printf("Entered String is %s\n",str);**

**//Storing string in terms of ascii and to restore spaces I used -20**

**size=strlen(str);**

**for(i=0;i<size;i++)**

**{**

**if(str[i]!=' ')**

**numstr[i]=str[i]-'A';**

**}**

**lennumstr=i;**

**//Key processing**

**printf("Enter the key (Non repeated elements if possible)\n");**

**gets(key);**

**//converting entered key to Capital letters**

**for(i=0,j=0;i<strlen(key);i++)**

**{**

**if(key[i]!=' ')**

**{**

**key[j]=toupper(key[i]);**

**j++;**

**}**

**}**

**key[j]='\0';**

**printf("%s\n",key);**

**//Storing key in terms of ascii**

**k=0;**

**for(i=0;i<strlen(key)+26;i++)**

**{**

**if(i<strlen(key))**

**{**

**if(key[i]=='J')**

**{**

**flag=8;**

**printf("%d",flag);**

**}**

**numkey[i]=key[i]-'A';**

**}**

**else**

**{**

**if(k!=9 && k!=flag)//Considering I=J and taking I in place of J except when J is there in key ignoring I**

**{**

**numkey[i]=k;**

**}**

**k++;**

**}**

**}**

**templen=i;**

**lenkey=removerepeated(templen,numkey);**

**printf("Entered key converted according to Play Fair Cipher rule\n");**

**for(i=0;i<lenkey;i++)**

**{**

**printf("%c",numkey[i]+'A');**

**}**

**printf("\n");**

**//Arranging the key in 5x5 grid**

**k=0;**

**for(i=0;i<5;i++)**

**{**

**for(j=0;j<5;j++)**

**{**

**cipherkey[i][j]=numkey[k];**

**k++;**

**}**

**}**

**printf("Arranged key\n");**

**for(i=0;i<5;i++)**

**{**

**for(j=0;j<5;j++)**

**{**

**printf("%c ",cipherkey[i][j]+'A');**

**}**

**printf("\n");**

**}**

**//Message Processing**

**for(i=0;i<lennumstr;i+=2)**

**{**

**if(numstr[i]==numstr[i+1])**

**{**

**insertelementat(i+1,numstr,lennumstr);**

**lennumstr++;**

**}**

**}**

**if(lennumstr%2!=0)**

**{**

**insertelementat(lennumstr,numstr,lennumstr);**

**lennumstr++;**

**}**

**printf("Entered String/Message After Processing according to Play fair cipher rule\n");**

**for(i=0;i<lennumstr;i++)**

**{**

**printf("%c",numstr[i]+'A');**

**}**

**for(k=0;k<lennumstr;k+=2)**

**{**

**for(i=0;i<5;i++)**

**{**

**for(j=0;j<5;j++)**

**{**

**if(numstr[k]==cipherkey[i][j])**

**{**

**row1=i;**

**col1=j;**

**}**

**if(numstr[k+1]==cipherkey[i][j])**

**{**

**row2=i;**

**col2=j;**

**}**

**}**

**}**

**//Only change between Ecryption to decryption is changing + to -**

**//If negative add 5 to that row or column**

**if(row1==row2)**

**{**

**col1=(col1-1)%5;**

**col2=(col2-1)%5;**

**if(col1<0)**

**{**

**col1=5+col1;**

**}**

**if(col2<0)**

**{**

**col2=5+col2;**

**}**

**numcipher[k]=cipherkey[row1][col1];**

**numcipher[k+1]=cipherkey[row2][col2];**

**}**

**if(col1==col2)**

**{**

**row1=(row1-1)%5;**

**row2=(row2-1)%5;**

**if(row1<0)**

**{**

**row1=5+row1;**

**}**

**if(row2<0)**

**{**

**row2=5+row2;**

**}**

**numcipher[k]=cipherkey[row1][col1];**

**numcipher[k+1]=cipherkey[row2][col2];**

**}**

**if(row1!=row2&&col1!=col2)**

**{**

**numcipher[k]=cipherkey[row1][col2];**

**numcipher[k+1]=cipherkey[row2][col1];**

**}**

**}**

**printf("\nCipher Text is\n");**

**for(i=0;i<lennumstr;i++)**

**{**

**if((numcipher[i]+'A')!='X')//Should remove extra 'X' which were created during Encryption**

**printf("%c",numcipher[i]+'A');**

**}**

**printf("\n");**

**}**

**int removerepeated(int size,int a[])**

**{**

**int i,j,k;**

**for(i=0;i<size;i++)**

**{**

**for(j=i+1;j<size;)**

**{**

**if(a[i]==a[j])**

**{**

**for(k=j;k<size;k++)**

**{**

**a[k]=a[k+1];**

**}**

**size--;**

**}**

**else**

**{**

**j++;**

**}**

**}**

**}**

**return(size);**

**}**

**int insertelementat(int position,int a[],int size)**

**{**

**int i,insitem=23,temp[size+1];**

**for(i=0;i<=size;i++)**

**{**

**if(i<position)**

**{**

**temp[i]=a[i];**

**}**

**if(i>position)**

**{**

**temp[i]=a[i-1];**

**}**

**if(i==position)**

**{**

**temp[i]=insitem;**

**}**

**}**

**for(i=0;i<=size;i++)**

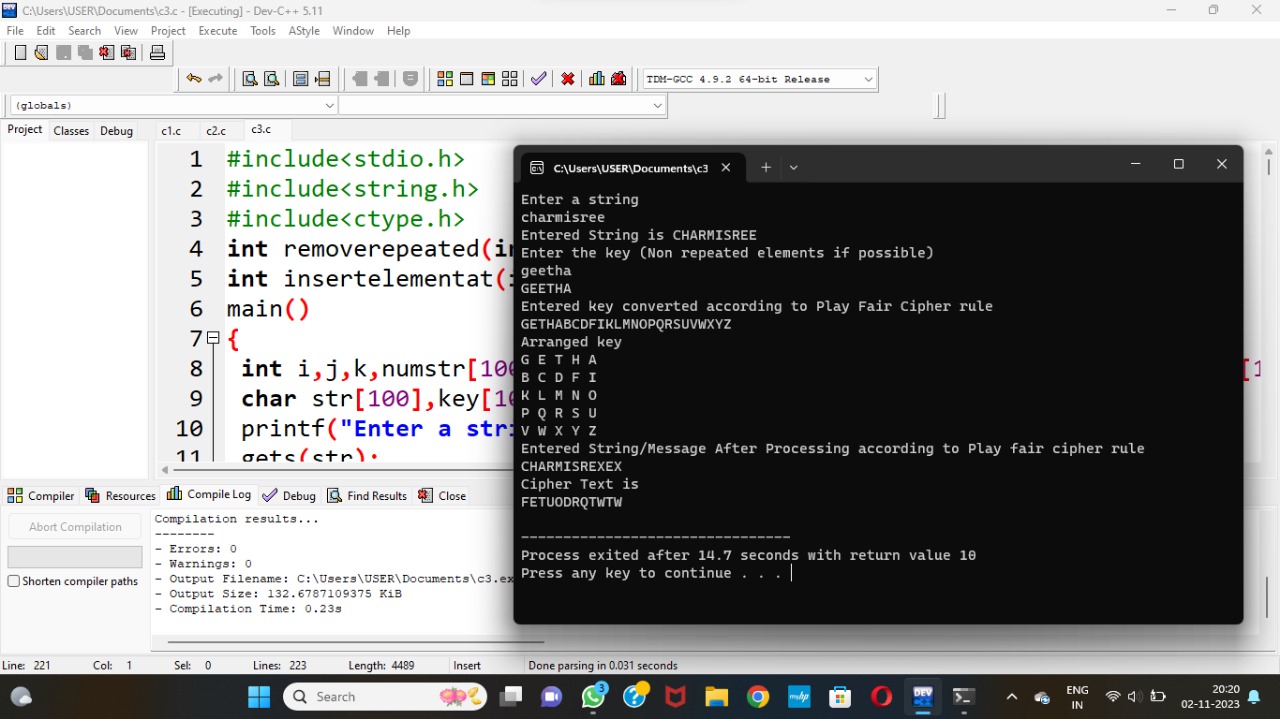
**{**

**a[i]=temp[i];**

**}**

**}**

**OUTPUT:-**

****

**4.Write a c program for polyalphabetic substitution cipher uses a separate monoalphabetic substitution cipher for each successive letter of plaintext, depending on a key.**

**PROGRAM:-**

**#include<stdio.h>**

**#include<conio.h>**

**#include<string.h>**

**Char pt[40]={‘\0’},key[40]={‘\0’},ct[40]={‘\0’},pta[40]={‘\0’},k[40]={‘\0’};**

**Int I,j; // global values**

**Int main()**

**{**

**Printf(“\nEnter the keyword:\n”);**

**Gets(k);//read the key**

**Printf(“\nEnter the Plain text:\n”);**

**Gets(pt); //read the plain text**

**// print the table**

**Printf(“The convergen matrix\n”);**

**Printf(“\n “);**

**For(j=97;j<=122;j++)**

**{**

**Printf(“ %c”,j);**

**}**

**Printf(“\n--------------------------------------------------------\n”);**

**For(i=97;i<=122;i++)**

**{**

**Printf(“%c |”,i);**

**For(j=97;j<=122;j++)**

**{**

**If(((i+j))>219)**

**{**

**Printf(“ %c”,toupper((i+j)-123));**

**}**

**Else**

**{**

**Printf(“ %c”,toupper((i+j)-97));**

**}**

**}**

**Printf(“\n”);**

**}**

**// for keyword**

**J=0;**

**For(i=0;i<strlen(pt);i++)**

**{**

**Key[i]=k[j];**

**If(j==(strlen(k)-1))**

**{**

**J=0;**

**}**

**Else**

**{**

**J++;**

**}**

**}**

**For(i=0;i<(strlen(pt)-1);i++);**

**K[i]=’\0’;**

**Printf(“\nThe encrypted text is:\n”);**

**// encryption**

**For(i=0;i<strlen(pt);i++)**

**{**

**If(97<=(int)pt[i] && (int)pt[i]<=122)**

**{**

**If(((int)pt[i]+(int)key[i])>219)**

**{**

**Ct[i]=(int)pt[i]+(int)key[i]-123;**

**}**

**Else**

**{**

**Ct[i]=(int)pt[i]+(int)key[i]-97;**

**}**

**Printf(“%c”,toupper(ct[i]));**

**}**

**Else**

**{**

**Ct[i]=pt[i];**

**Printf(“%c”,pt[i]);**

**}**

**}**

**// decryption**

**Printf(“\n\nDecrypted text is:\n”);**

**For(i=0;i<strlen(ct);i++)**

**{**

**If(97<=(int)ct[i] && (int)ct[i]<=122)**

**{**

**If(((int)ct[i]-(int)key[i])<0)**

**{**

**Pta[i]=((int)ct[i]-(int)key[i])+123;**

**}**

**Else**

**{**

**Pta[i]=(int)ct[i]-(int)key[i]+97;**

**}**

**Printf(“%c”,pta[i]);**

**}**

**Else**

**{**

**Pta[i]=ct[i];**

**Printf(“%c”,pta[i]);**

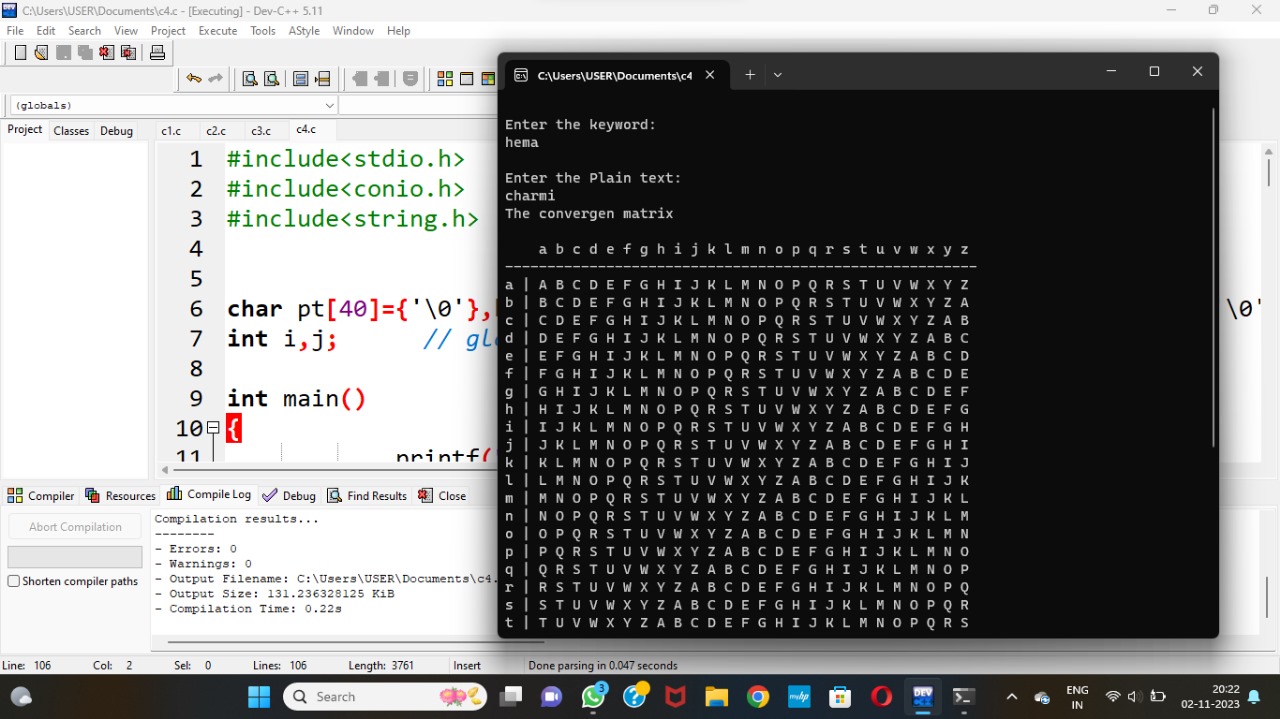
**}**

**}**

**Return 0;**

**}**

**OUTPUT:-**

****

**5. Write a C program forgeneralization of the Caesar cipher, known as the affine Caesar**

**Cipher, has thefollowing form: For each plaintext letter p, substitute the**

**ciphertextletterC: C = E([a, b], p) = (ap + b) mod 26 A basic requirement of**

**any encryption algorithm is that it be one-to-one. That is, if p q,then E(k, p)**

**E(k, q). Otherwise, decryption is impossible, because more than one plaintext**

**Character maps into the same ciphertext character. The affine Caesar cipher is**

**Not one-to one for all values of a. For example, for a = 2 and b = 3,then**

**E([a,b], 0) = E([a, b], 13) = 3.**

**a.Are there any limitations on the value of b?**

**b.Determine which values of a are not allowed.**

**PROGRAM:-**

**#include <stdio.h>**

**Int gcd(int a, int b) {**

**If (b == 0) {**

**Return a;**

**}**

**Return gcd(b, a % b);**

**}**

**Int main() {**

**Int a;**

**Printf(“Allowed values of ‘a’ for the affine Caesar cipher: “);**

**For (a = 1; a < 26; a++) {**

**If (gcd(a, 26) == 1) {**

**Printf(“%d “, a);**

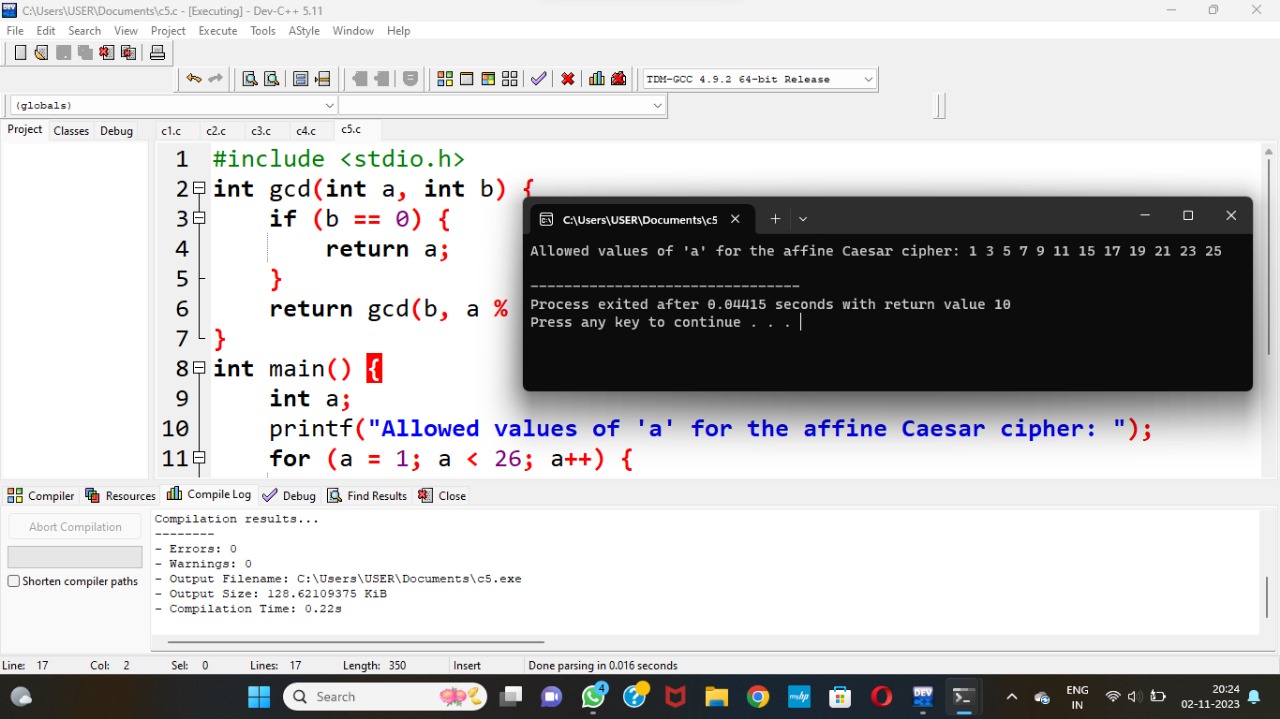
**}**

**}**

**Printf(“\n”);**

**}**

**OUTPUT:-**

****

**6. Write a C program for cipher text has been generated with an affine cipher most**

**Frequent letter of the ciphertext is “B,” and the second most**

**Frequent letter of the ciphertext is “U.” Break this code.**

**PROGRAM:-**

**#include <stdio.h>**

**Int main() {**

**Char ciphertext[] = “BUWXYBZVBQBUCB”;**

**Char mostFrequent = ‘B’;**

**Char secondMostFrequent = ‘U’;**

**Int mostFrequentNum = mostFrequent – ‘A’;**

**Int secondMostFrequentNum = secondMostFrequent – ‘A’;**

**Int a, b, I;**

**For (a = 1; a < 26; a++) {**

**Int candidateB = (secondMostFrequentNum – (a \* mostFrequentNum) + 26) % 26;**

**Int valid = 1,I;**

**For (I = 0; I < sizeof(ciphertext) – 1; i++) {**

**Int ciphertextNum = ciphertext[i] – ‘A’;**

**Int decryptedNum = (a \* ciphertextNum + candidateB) % 26;**

**If (decryptedNum < 0) {**

**Valid = 0;**

**Break;**

**}**

**}**

**If (valid) {**

**B = candidateB;**

**Break;**

**}**

**}**

**Printf(“Decrypted message: “);**

**For (I = 0; ciphertext[i] != ‘\0’; i++) {**

**Int ciphertextNum = ciphertext[i] – ‘A’;**

**Int decryptedNum = (a \* ciphertextNum + b) % 26;**

**Char decryptedChar = decryptedNum + ‘A’;**

**Printf(“%c”, decryptedChar);**

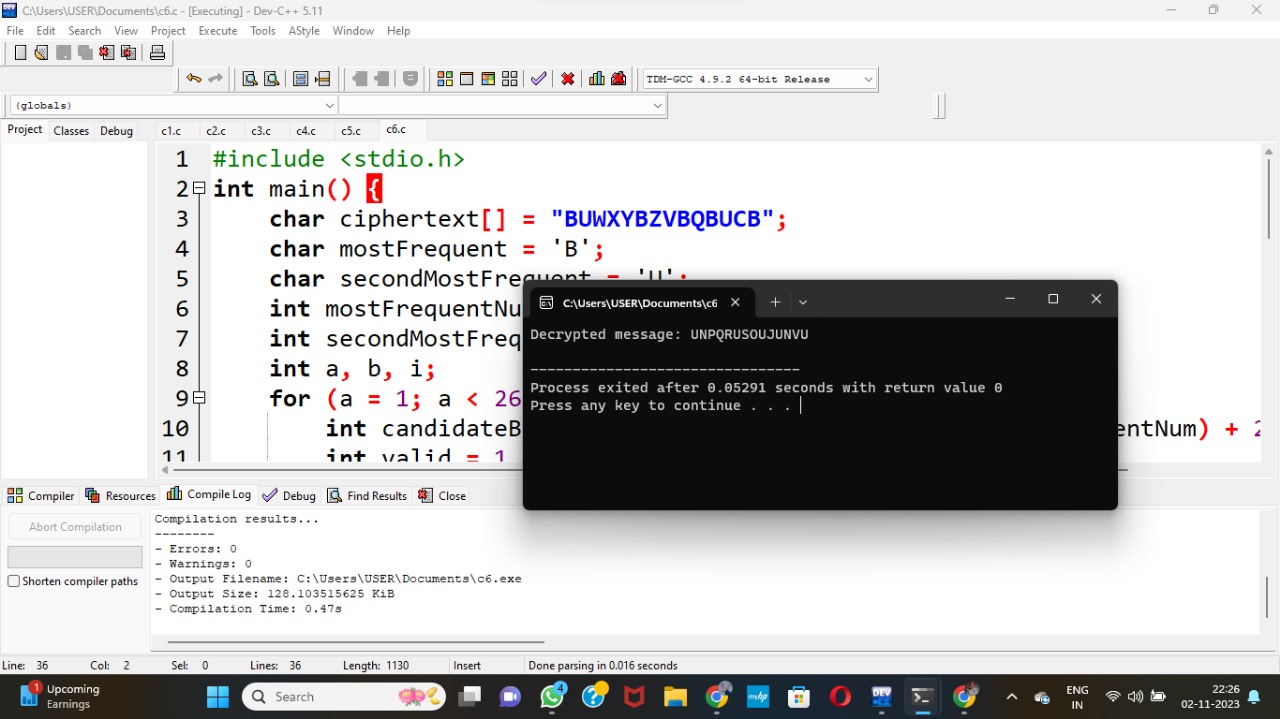
**}**

**Printf(“\n”);**

**Return 0;**

**}**

**OUTPUT:-**

****