PROGRAMS

6. Break the following ciphertext “PELCGBTENCULVFSBEVASBEZNGVBAFRPHEVGL” which is generated by monoalphabetic additive substitution cipher.

Python code

```
import string
all_letters= string.ascii_letters
dict1 = {}
cipher_txt= "PELCGBTENCULVFSBEVASBEZNGVBAFRPHEVGL"
print("Cipher Text is: ",cipher_txt)
for key in range(26):
    dict2 = {}
    for i in range(len(all_letters)):
        dict2[all_letters[i]] = all_letters[(i-key)%(len(all_letters))]
    decrypt_txt = []
    for char in cipher_txt:
        if char in all_letters:
            temp = dict2[char]
            decrypt_txt.append(temp)
        else:
            temp = char
            decrypt_txt.append(temp)
    decrypt_txt = "".join(decrypt_txt)
    print("Recovered plain text :", decrypt_txt)
```

Output:

Cipher Text is: PELCGBTENCULVFSBEVASBEZNGVBAFRPHEVGL

Recovered plain text : PELCGBTENCULVFSBEVASBEZNGVBAFRPHEVGL

Recovered plain text : ODKBFASDMBTKUERADUzRADYMFUAzEQOGDUFK

Recovered plain text : NCJAEzRCLASJTDQzCTyQzCXLETzyDPNFCTEJ

Recovered plain text : MBIzDyQBKzRISCPyBSxPyBWKDSyxCOMEBSDI

Recovered plain text : LAHyCxPAJyQHRBOxARwOxAVJCRxwBNLDARCH

Recovered plain text : KzGxBwOzIxPGQANwzQvNwzUIBQwvAMKCzQBG

Recovered plain text : JyFwAvNyHwOFPzMvyPuMvyTHAPvuzLJByPAF

Recovered plain text : IxEvzuMxGvNEOyLuxOtLuxSGzOutyKIAxOzE

Recovered plain text : HwDuytLwFuMDNxKtwNsKtwRFyNtsxJHzwNyD

Recovered plain text : GvCtxsKvEtLCMwJsvMrJsvQExMsrwIGyvMxC

Recovered plain text : FuBswrJuDsKBLvIruLqIruPDwLrqvHFxuLwB

Recovered plain text : EtArvqItCrJAKuHqtKpHqtOCvKqpuGEwtKvA

Recovered plain text : DszqupHsBqIzJtGpsJoGpsNBuJpotFDvsJuz

Recovered plain text : CryptoGrApHyIsForInForMAtIonsECurIty

Recovered plain text : BqxosnFqzoGxHrEnqHmEnqLzsHnmrDBtqHsx

Recovered plain text : ApwnrmEpynFwGqDmpGlDmpKyrGmlqCAspGrw

Recovered plain text : zovmqlDoxmEvFpCloFkCloJxqFlkpBzroFqv

Recovered plain text : ynulpkCnwlDuEoBknEjBknIwpEkjoAyqnEpu

Recovered plain text : xmtkojBmvkCtDnAjmDiAjmHvoDjinzxpmDot

Recovered plain text : wlsjniAlujBsCmzilChzilGunCihmywolCns

Recovered plain text : vkrimhzktiArBlyhkBgyhkFtmBhglxvnkBmr

Recovered plain text : ujqhlgyjshzqAkxgjAfxgjEslAgfkwumjAlq

Recovered plain text : tipgkfxirgypzjwfizewfiDrkzfejvtlizkp

Recovered plain text : shofjewhqfxoyivehydvehCqjyediuskhyjo

Recovered plain text : rgneidvgpewnxhudgxcudgBpixdchtrjgxin

Recovered plain text : qfmdhcufodvmwgtcfwbtcfAohwcbgsqifwhm

Java code

```
import java.io.*;
public class AdditiveCipher {
   public static void main(String [] args )
   {
      char alpha[] = {'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U',
'V', 'W', 'X', 'Y', 'Z'};
      String s = "PELCGBTENCULVFSBEVASBEZNGVBAFRPHEVGL";
      char[] a1 = s.toCharArray();
      int c = 0, f = 0;
      double e = 0, a = 0;
      String val = " ";
      for (int i = 0; i <26; i++)
      {
         val=" ";
         System.out.print("For K " + i);
         for (int a_r = 0; a_r < s.length(); a_r++)
         {
            for (int j = 0; j <26; j++)
            {
               if (a1[a_r] == alpha[j])
               {
```

```
c = j;
a = c - i;



if (a < 0)
{
   a = a / 26; a = a + 10;
   e = a;
   if (a >=1) {
      e = a - Math.floor(a);
      e = e * 26;
   } else {
      e = e * 26;
   }

} else
   {
   a = a / 26;
   e = a;
   if (a >=1) {
      e = a - Math.floor(a);
      e = e * 26;
   } else {
      e = e * 26;
   }}
 int k = (int) e;
```

```
                val = val + alpha[k];
              }
            }
          }
          System.out.println(val);
      }
  }
  }
```

Output:

For K 0OELCGBTENCULVFSBEVASBEZNGVBAFROHEVGL

For K 1ODKBFASDMBTKUERADUZRADYMFUAZEQOGDUFK

For K 2 NCJAEZRCLASJTDQZCTYQZCXLETZYDONFCTEJ

For K 3MBIZDYQBKZRISCOYBSXOYBWKDSYXCOMEBSDI

For K 4LAHYCXOAJYQHRBOXARWOXAVJCRXWBNLDARCH

For K 5KZGXBWOZIXOGQANWZQVNWZUIBQWVAMKCZQBG

For K 6JYFWAVNYHWOFOZMVYOUMVYTHAOVUZLJBYOAF

For K 7IXEVZUMXGVNEOYLUXOSLUXSGZOUSYKIAXOZE

For K 8HWDUYSLWFUMDNXKSWNRKSWRFYNSRXJHZWNYD

For K 9GVCSXRKVESLCMWJRVMQJRVQEXMRQWIGYVMXC

For K 10 FUBRWQJUDRKBLVIQULPIQUODWLQPVHFXULWB

For K 11 ESAQVPISCQJAKUHPSKOHPSOCVKPOUGEWSKVA

For K 12 DRZPUOHRBPIZJSGORJNGORNBUJONSFDVRJUZ

For K 13 CQYOSNGQAOHYIRFNQINFNQMASINNRECUQISY

For K 14 BPXNRNFPZNGXHQENPHMENPLZRHNMQDBSPHRX

For K 15 AOWNQMEOYNFWGPDMOGLDMOKYQGMLPCAROGQW

For K 16 ZNVMPLDNXMEVFOCLNFKCLNJXPFLKOBZQNFPV

For K 17 YNULOKCNWLDUENBKNEJBKNIWOEKJNAYPNEOU

For K 18 XMSKNJBMVKCSDNAJMDIAJMHVNDJINZXOMDNS

For K 19 WLRJNIALUJBRCMZILCHZILGUNCIHMYWNLCNR

For K 20 VKQIMHZKSIAQBLYHKBFYHKFSMBHFLXVNKBMQ

For K 21 UJPHLFYJRHZPAKXFJAEXFJERLAFEKWUMJALP

For K 22 SIOFKEXIQFYOZJWEIZDWEIDQKZEDJVSLIZKO

For K 23RHNEJDWHPEXNYIVDHYCVDHCPJYDCIURKHYJN

For K 24QFNDICVFODWNXHUCFXBUCFBOIXCBHSQJFXIN

For K 25PEMCHBUENCVMWFSBEWASBEANHWBAFRPIEWHM

7. Break the ciphertext “UNTWXEAPUWNUGGKSYXK” which is generated by monoalphabetic multiplicative substitution cipher

```
import java.io.*;

public class Multiplicative {

   public static void main(String [] args )

   {

      int c=0,b=0;

      double e=0;

      long a=0;

      double e1;

      String s = "UNTWXEAPUWNUGGKSYXK";

      char alpha[] = {'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U',
'V', 'W', 'X', 'Y', 'Z'};

      char[] a1 = s.toCharArray();

      String val=" ";

      for (int i = 0; i <26; i++)
```

```
{
   val=" ";
   System.out.print("When K " + i);
    a= cipher(26,i);
   if(a<0)
   {
      a=a+26;
   }
    for (int a_r = 0; a_r < s.length(); a_r++)
   {
       for (int j = 0; j < 26; j++)
      {

         if (a1[a_r] == alpha[j])
         {
            b=j;
            e=b*(int)a;
          e1=e/26;
          if(e1>=1)
          {
             e1=e1-Math.floor(e1);
             e1=e1*26;
          }else
          {
              e1=e1*26;
          }
```

```
                int k=(int)e1;
                 val=val+alpha[k];
              }
           }
        }
        System.out.println(val);
        } }
  public static long cipher(long a, long b)
  {
     long x = 0, y = 1, lastx = 1, answer = 0, temp;
     while (b != 0)
     {
        long q = a / b;
        long r = a % b;
        a = b;b = r;
        temp = x;
        x = lastx - q * x;
        lastx = temp;

        temp = y;
        y = answer - q * y;
        answer = temp;
     }
    return answer;
  } }
```

Output:

When K 1 UNTWXEAOUWNUGGKSYXK

When K 2UNTWXEAOUWNUGGKSYXK

When K 3YNOPZKAFYPNYCCMGIZM

When K 4 KAPYSCANKYAKPPSWLSS

When K 5DNJUOGACDUNDVVBNJOB

When K 6YABQLKARYQAYBBMFHLM

When K 7NNZRHHAQNRNNMMUKWHU

When K 8SNULJNAHSLNSHHWYFJW

When K 9HNFNQMATHNNHSSDCUQD

When K 10DNJUOGACDUNDVVBNJOB

When K 11PNXBUXAZPBNPKKHDOUH

When K 12LAOHFSAWLHALNNFQDFF

When K 13UNTWXEAOUWNUGGKSYXK

When K 14 OALSUIADOSAOMMUKWUU

When K 15 KNCYFBAAKYNKQQSVMFS

When K 16 WNQGLUAXWGNWDDYMPLY

When K 17 SNULJNAHSLNSHHWYFJW

When K 18 HNFNQMATHNNHSSDCUQD

When K 19 MNAISSAIMINMNNGPDSG

When K 20 CAXKNQAHCKACYYOUSNO

When K 21 WNQGLUAXWGNWDDYMPLY

When K 22 PAKBHYAMPBAPKKHENHH

When K 23 BNLKAQAVBKNBXXNURAN

When K 24 FNHDDWALFDNFUUPHBDP

When K 25 FNHDDWALFDNFUUPHBDP

10. Break the ciphertext “MTMTCMSALHRDY” which is generated using Auto-key cipher.

Java code

```
import java.lang.*;
import java.util.*;
public class AutoKey {
	private static final String alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
	public static void main(String[] args)
	{
		String enc = "MTMTCMSALHRDY";
		for(int i=0;i<=26;i++){
		   int key1 = i;
		String key = String.valueOf(key1);
		if (key.matches("[-+]?\\d*\\.?\\d+"))
			key = "" + alphabet.charAt(Integer.parseInt(key));
			System.out.println("key "+key);
		System.out.println("Encrypted : " + enc);
		System.out.println("Decrypted : " + autoDecryption(enc, key));
		}
	}
	public static String autoDecryption(String msg, String key)
	{
		String currentKey = key;
		String decryptMsg = "";
		for (int x = 0; x < msg.length(); x++) {
```

```
                int get1 = alphabet.indexOf(msg.charAt(x));
                int get2 = alphabet.indexOf(currentKey.charAt(x));
                int total = (get1 - get2) % 26;
                total = (total < 0) ? total + 26 : total;
                decryptMsg += alphabet.charAt(total);
                currentKey += alphabet.charAt(total);
            }
            return decryptMsg;
        }
}
```

Output:

```
$javac AutoKey.java
$java -Xmx128M -Xms16M AutoKey
key A
Encrypted : MTMTCMSALHRDY
Decrypted : MHFOOYUGFCPOK
key B
Encrypted : MTMTCMSALHRDY
Decrypted : LIEPNZTHEDOPJ
key C
Encrypted : MTMTCMSALHRDY
Decrypted : KJDQMASIDENQI
key D
Encrypted : MTMTCMSALHRDY
Decrypted : JKCRLBRJCFMRH
key E
Encrypted : MTMTCMSALHRDY
Decrypted : ILBSKCQKBGLSG
key F
Encrypted : MTMTCMSALHRDY
Decrypted : HMATJDPLAHKTF
key G
Encrypted : MTMTCMSALHRDY
Decrypted : GNZUIEOMZIJUE
key H
Encrypted : MTMTCMSALHRDY
```

```
Decrypted : FOYVHFNNYJIVD
key I
Encrypted : MTMTCMSALHRDY
Decrypted : EPXWGGMOXKHWC
key J
Encrypted : MTMTCMSALHRDY
Decrypted : DQWXFHLPWLGXB
key K
Encrypted : MTMTCMSALHRDY
Decrypted : CRVYEIKQVMFYA
key L
Encrypted : MTMTCMSALHRDY
Decrypted : BSUZDJJRUNEZZ
key M
Encrypted : MTMTCMSALHRDY
Decrypted : ATTACKISTODAY
key N
Encrypted : MTMTCMSALHRDY
Decrypted : ZUSBBLHTSPCBX
key O
Encrypted : MTMTCMSALHRDY
Decrypted : YVRCAMGURQBCW
key P
Encrypted : MTMTCMSALHRDY
Decrypted : XWQDZNFVQRADV
key Q
Encrypted : MTMTCMSALHRDY
Decrypted : WXPEYOEWPSZEU
key R
Encrypted : MTMTCMSALHRDY
Decrypted : VYOFXPDXOTYFT
key S
Encrypted : MTMTCMSALHRDY
Decrypted : UZNGWQCYNUXGS
key T
Encrypted : MTMTCMSALHRDY
Decrypted : TAMHVRBZMVWHR
key U
Encrypted : MTMTCMSALHRDY
Decrypted : SBLIUSAALWVIQ
key V
Encrypted : MTMTCMSALHRDY
Decrypted : RCKJTTZBKXUJP
```

```
key W
Encrypted : MTMTCMSALHRDY
Decrypted : QDJKSUYCJYTKO
key X
Encrypted : MTMTCMSALHRDY
Decrypted : PEILRVXDIZSLN
key Y
Encrypted : MTMTCMSALHRDY
Decrypted : OFHMQWWEHARMM
key Z
Encrypted : MTMTCMSALHRDY
Decrypted : NGGNPXVFGBQNL
```