**Question 1**

1. import java.util.\*;
2. public class Multi
3. {
4. public static void main(String args[])
5. {
6. Scanner sc=new Scanner(System.in);
7. System.out.println("Enter 1 for encryption, 2 for decryption");
8. int choice=sc.nextInt();
9. sc.nextLine();
10. int Z=26;
11. String text,s;
12. int key,length;
13. int i,j,dup1,dup2;
14. char ch;
15. switch(choice)
16. {
17. case 1:System.out.println("Enter plaintext");
18. text=sc.nextLine();
19. System.out.println("Enter key present in Z\*26");
20. key=sc.nextInt();
21. dup1=Z;
22. dup2=key;

25. //Calculating GCD
26. while(dup1!=dup2)
27. {
28. if(dup1<dup2)
29. {
30. dup2=dup2-dup1;
31. }
32. else if(dup2<dup1)
33. {
34. dup1=dup1-dup2;
35. }
36. }


40. //GCD stored in dup1 and dup2
41. if(dup1!=1)
42. {
43. //Encryption can be done but the resulting ciphertext cannot be decrypted back is gcd is not equal to 1
44. System.out.println("Decryption with this key will not be possible. Please enter a Integer present in Z\*26");
45. break;
46. }

49. length=text.length();
50. s=""; //String to store ciphertext
51. for(i=0;i<length;i++)
52. {
53. ch=text.charAt(i);
54. if((ch>='a')&&(ch<='z'))
55. {
56. ch=(char)(ch-'a');//subtracting ASCII value from character
57. ch=(char)((ch\*key)%26);// doing mod in case of overflow
58. ch=(char)(ch+'a');//adding ASCII value back to get corresponding character
59. s=s+ch;
60. }
61. else if((ch>='A')&&(ch<='Z'))
62. {
63. ch=(char)(ch-'A');//subtracting ASCII value from character
64. ch=(char)((ch\*key)%26);// doing mod in case of overflow
65. ch=(char)(ch+'A');//adding ASCII value back to get corresponding character
66. s=s+ch;
67. }
68. else
69. {
70. s=s+ch;
71. }
72. }
73. System.out.println("Ciphertext is "+s);
74. break;
76. case 2:System.out.println("Enter ciphertext");
77. text=sc.nextLine();
78. System.out.println("Enter key present in Z\*26");
79. key=sc.nextInt();
80. dup1=Z;
81. dup2=key;

84. //Calculating GCD
85. while(dup1!=dup2)
86. {
87. if(dup1<dup2)
88. {
89. dup2=dup2-dup1;
90. }
91. else if(dup2<dup1)
92. {
93. dup1=dup1-dup2;
94. }
95. }


99. //GCD stored in dup1 and dup2
100. if(dup1!=1)
101. {
102. System.out.println("Decryption with this key will not be possible. Please enter a Integer present in Z\*26");
103. break;
104. }
106. //Extended Euclidean Algorithm to calculate Multiplicative Inverse
107. int r1=Z;
108. int r2=key;
109. int t1=0;
110. int t2=1;
111. int q,t,r;
112. while(r2>0)
113. {
114. q=r1/r2;
115. r=r1-q\*r2;
116. r1=r2;
117. r2=r;
118. t=t1-q\*t2;
119. t1=t2;
120. t2=t;
121. if(r1==1)
122. {
123. while(t1<0)
124. {
125. t1=t1+Z;
126. }
127. }
128. }
129. //Multiplicative Inverse stored in t1
131. length=text.length();
132. s=""; //String to store plaintext
133. for(i=0;i<length;i++)
134. {
135. ch=text.charAt(i);
136. if((ch>='a')&&(ch<='z'))
137. {
138. ch=(char)(ch-'a');//subtracting ASCII value from character
139. ch=(char)((ch\*t1)%26);// doing mod in case of overflow
140. ch=(char)(ch+'a');//adding ASCII value back to get corresponding character
141. s=s+ch;
142. }
143. else if((ch>='A')&&(ch<='Z'))
144. {
145. ch=(char)(ch-'A');//subtracting ASCII value from character
146. ch=(char)((ch\*t1)%26);// doing mod in case of overflow
147. ch=(char)(ch+'A');//adding ASCII value back to get corresponding character
148. s=s+ch;
149. }
150. else
151. {
152. s=s+ch;
153. }
154. }
155. System.out.println("Plaintext is "+s);
156. break;
157. default:System.out.println("Enter correct choice");
158. }
159. }
160. }