

Code : 1

```
package Topic_10_RecursionOnTheWayUp;
```

```
import java.util.Scanner;
```

```
public class A_PrintSubSequence {  
  
    public static void main(String[] args) throws Exception {  
        Scanner scn = new Scanner(System.in);  
        String str = scn.next();  
        printSS(str, ""); //1  
    }  
  
    public static void printSS(String ques, String ans) {  
        if (ques.length() == 0) { //2  
            System.out.println(ans);  
            return;  
        }  
  
        char ch = ques.charAt(0); //3  
        String roq = ques.substring(1); //4  
        printSS(roq, ans + ch); //5  
        printSS(roq, ans + ""); //6  
    }  
}
```

Code : 2

```
package Topic_10_RecursionOnTheWayUp;
```

```
import java.io.*;
```

```
import java.util.*;
```

```
public class B_PrintKPC {
```

```
    public static void main(String[] args) throws Exception {  
        Scanner scn = new Scanner(System.in);  
        String str = scn.next();  
        printKPC(str, "");  
    }
```

```
    static String[] codes = { ".", "abc", "def", "ghi", "jkl", "mno", "pqrs", "tu", "vwxyz", "yz" }; //1
```

```
    public static void printKPC(String ques, String ans) {  
        if (ques.length() == 0) //2  
        {  
            System.out.println(ans);  
            return;  
        }  
        char ch = ques.charAt(0); //3  
        String roq = ques.substring(1); //4  
        String codeforch = codes[ch - 'a']; //5  
        for (int i = 0; i < codeforch.length(); i++) //6  
        {  
            char cho = codeforch.charAt(i);  
            printKPC(roq, ans + cho); //7  
        }  
    }
```

```
}
```

Code : 3

```
package Topic_10_RecursionOnTheWayUp;
```

```
import java.io.*;
```

```
import java.util.*;
```

```
public class C_PrintStairPaths {
```

```
    public static void main(String[] args) throws Exception {  
        Scanner scn = new Scanner(System.in);  
        int t = scn.nextInt();  
        printStairPaths(t, "");  
    }
```

```
    public static void printStairPaths(int n, String psf) {  
        if (n <= 0) {  
            if (n == 0) {  
                System.out.println(psf);  
            }  
  
            return;  
        }
```

```
        printStairPaths(n - 1, psf + 1);  
        printStairPaths(n - 2, psf + 2);  
        printStairPaths(n - 3, psf + 3);
```

```
    }
```

```
}
```

Code : 4

```
package Topic_10_RecursionOnTheWayUp;
```

```
import java.util.*;
```

```
public class D_PrintMazePath {
```

```
    public static void main(String[] args) throws Exception {  
        Scanner scn = new Scanner(System.in);  
        int n = scn.nextInt();  
        int m = scn.nextInt();  
        printMazePaths(0, 0, n - 1, m - 1, "");  
    }
```

```
    public static void printMazePaths(int sr, int sc, int dr, int dc, String psf) {  
        if (sr > dr || sc > dc) {  
            return;  
        }  
  
        if (sr == dr && sc == dc) {  
            System.out.println(psf);  
            return;  
        }  
  
        printMazePaths(sr, sc + 1, dr, dc, psf + "h");  
        printMazePaths(sr + 1, sc, dr, dc, psf + "v");  
    }
```

```
}
```

Code : 5

```
package Topic_10_RecursionOnTheWayUp;
```

```
import java.io.*;
```

```
public class E_PrintMazePathsWithJumps {
```

```
    public static void main(String[] args) throws Exception {  
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));  
        int n = Integer.parseInt(br.readLine());  
        int m = Integer.parseInt(br.readLine());  
        printMazePaths(0, 0, n - 1, m - 1, "");  
    }
```

```
    public static void printMazePaths(int sr, int sc, int dr, int dc, String psf) {
```

```
        if (sr == dr && sc == dc) {  
            System.out.println(psf);  
            return;  
        }
```

```
        for (int move = 1; move <= dc - sc; move++) {  
            printMazePaths(sr, sc + move, dr, dc, psf + "h" + move);  
        }
```

```
        for (int move = 1; move <= dr - sr; move++) {  
            printMazePaths(sr + move, sc, dr, dc, psf + "v" + move);  
        }
```

```
        for (int move = 1; move <= dc - sc && move <= dr - sr; move++) {  
            printMazePaths(sr + move, sc + move, dr, dc, psf + "d" + move);  
        }
```

```
    }
```

```
}
```

Code : 6

```
package Topic_10_RecursionOnTheWayUp;
```

```
import java.util.*;
```

```
public class F_PrintPermutations {
```

```
    public static void main(String[] args) throws Exception {
```

```
        Scanner scn = new Scanner(System.in);
```

```
        String str = scn.next();
```

```
        printPermutations2(str, "");
```

```
    }
```

```
    private static void printPermutations2(String str, String ans) {
```

```
        if (str.length() == 0) {
```

```
            System.out.println(ans);
```

```
            return;
```

```
        }
```

```
        for (int i = 0; i < str.length(); i++) {
```

```
            char ch = str.charAt(i);
```

```
            StringBuilder s = new StringBuilder(str);
```

```
            s.deleteCharAt(i);
```

```
            printPermutations2(s.toString(), ans + ch);
```

```
        }
```

```
    }
```

```
    public static void printPermutations1(String str, String asf) {
```

```
        if (str.length() == 0) {
```

```
            System.out.println(asf); //Question string is empty so print the answer now and return  
            return;
```

```
        }
```

```
        //Extracting each character at a time from the question string and appending it to answer so far
```

```
        for (int i = 0; i < str.length(); i++) {
```

```
            char ch = str.charAt(i);
```

```
            String leftPart = str.substring(0, i); //Substring from 0 to i-1 (left to ch)
```

```
            String rightPart = str.substring(i + 1); //Substring from i+1 till end of String (right to ch)
```

```
            String roq = leftPart + rightPart; //Remaining string after extracting ch
```

```
            printPermutations1(roq, asf + ch);
```

```
        }
```

```
    }
```

```
}
```

Code : 7

```
package Topic_10_RecursionOnTheWayUp;
```

```
import java.io.*;
```

```
public class G_PrintEncodings {
```

```
    public static void main(String[] args) throws Exception {  
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));  
        String str = br.readLine();  
        printEncodings(str, "");  
    }
```

```
    public static void printEncodings(String ques, String ans) {  
        if (ques.length() == 0) {  
            System.out.println(ans);  
            return;  
        } else if (ques.length() == 1) {  
            if (ques.charAt(0) == '0') {  
                return;  
            } else {  
                String ch0 = ques.substring(0, 1);  
                String roq0 = ques.substring(1);  
                String code0 = (char) ('a' + (Integer.parseInt(ch0) - 1)) + "";  
                printEncodings(roq0, ans + code0);  
            }  
        } else {  
            if (ques.charAt(0) == '0') {  
                return;  
            } else {  
                String ch0 = ques.substring(0, 1);  
                String roq0 = ques.substring(1);  
                String code0 = (char) ('a' + (Integer.parseInt(ch0) - 1)) + "";  
                printEncodings(roq0, ans + code0);  
  
                String ch01 = ques.substring(0, 2);  
                String roq01 = ques.substring(2);  
                String code01 = (char) ('a' + (Integer.parseInt(ch01) - 1)) + "";  
  
                if (Integer.parseInt(ch01) <= 26) {  
                    printEncodings(roq01, ans + code01);  
                }  
            }  
        }  
    }  
}
```

