```
/* Ipsy Code for Data Engineer Position */
/*Contract : the function in the code should return true or false regarding
whether the person can cross the river */
/*Purpose : the code takes a string of asterisk with gaps and return whether
such a pattern of stones can help a person cross the river*/
/* Approach Used: Dynamic Programming */
/* time complexity of the approach is O(n^2)*/

import java.lang.Math; // Math class is required for sqrt function

public class RiverCrossing
{
public static boolean canCrossRiver(int array[], int rlength)
{
      int i=0,j=0; //i still took this case as I did not had any
//time constraints


    /*Base case*/
    if (rlength==0)
    return false;

      int jumps=(int)Math.sqrt(2*rlength); // maximum number of //jumps the that
can be there, person will take for crossing the
      //river, i used it just for intializing the space of the table //in 2D
array, its an approximation

 //reachability is used for finding whether any next step should be possible or
not, if a step is possible it will get a value of 1 otherwise 0

      int [][]reachability =  new int [rlength][jumps+1];
      //Initializing reachability to 0
     for(i=0;i<rlength;i++)
      {
           for(j=0;j<jumps;j++)
           {
               reachability[i][j]=0;
           }
      }

      // Initializing first row with all the jumps value equal to 0,
//initializing all the values in the length of the string with initial
      //jump(speed) equal to 0.
      for(j=0;j<=jumps;j++)
      {
           reachability[0][j]=0;
      }
      for(i=1;i<rlength;i++)
      {
           reachability[i][0]=0;
      }

      reachability[0][1] = array[0];
      /* dynamic programming structure */
```

```
      for(i=1;i<rlength;i++)
      {
           for(j=1;j<=jumps;j++)
            {
               if((array[i]==1) && (i>=j))
                {//Person came to (i-j)th location, with last speed of  j, so
it
                     //could take jth speed  by rule of equal speed
                     if(reachability[i-j][j]==1)
                     {
                           reachability[i][j]=1;
                     }
//Person came to (i-j)th location, with last speed of  j+1, so it //could take
jth speed  by rule of -1
                 else if ((j+1 < jumps) && (reachability[i-j][j+1]==1))
                      {
                           reachability[i][j]=1;
                     }
//person came to (i-j)th location, with last speed of  j-1, so again //it could
take jth speed by rule of +1
                     else if ((reachability[i-j][j-1]==1) && j>0)
                     {
                           reachability[i][j]=1;
                     }
                }
//If person is at ith location and speed is less than j+1, so can //cross river
itself
                      if((i+j+1 >= rlength) && (reachability[i][j]==1))
                     {
                     return true;
                     }
                 }
                 }
      return false;
}


public static void main(String args[])
{
      /* for simplicity I have converted '*' to 1 and ' ' to 0*/

      String str="*****  *   * * *  *  *  ";
      int len=str.length();
      int []arr=new int [len];
    int i=0;
    for(i=0;i<len;i++)//loop for converting string into 0s and 1s
    {
           if (str.charAt(i) == '*')
              arr[i] = 1;
           else
              arr[i] = 0;
    }
         System.out.println(canCrossRiver(arr,len));
}}
```