Implement the Rabin-Karp algorithm for substring search using a rolling hash. Discuss the impact of hash collisions on the algorithm's performance and how to handle them.

```
public class RabinKarpAlgorithm {
    private final int d=256;
    private final int q=101;

    void search(String pattern,String text) {
        int m= pattern.length();
        int n= text.length();
        int p=0;
        int t=0;
        int h=1;

        for(int i=0; i<m-1; i++) {
            h=(h*d) % q;
        }

        for(int i=0; i<m; i++) {
            p= (d*p + pattern.charAt(i)) % q;
            t= (d*t + text.charAt(i)) % q;
        }
        for(int i=0; i<n-m; i++) {
            if(p == t) {
                boolean match = true;
                for(int j=0; j<m; j++) {
                    if(text.charAt(i+j) != pattern.charAt(j)) {
                        match =  false;
                    break;
                    }
                }
```

```
                    if(match) {
                            System.out.println("Pattern found at index : " +  i);
                    }
            }
            if(i< n-m) {
        t= (d*(t-text.charAt(i) * h) + text.charAt(i+m)) %q;

        if(t<0) {
             t= (t+q);
        }
            }
    }
    }
            public static void main(String[] args) {
                    // TODO Auto-generated method stub
      String text ="ABBCCDDAEFG";
      String pattern ="BCC";
      RabinKarpAlgorithm obj= new RabinKarpAlgorithm();
      obj.search(pattern, text);
            }

}
```