Problem I
Disjoint uni-color paths
Input File: *pi.in*
Time Limit: *2 Seconds*

Solution
The main steps of the algorithm are as follows.

1. First, any edge between s, t must be a solution. Note that there may be many st-edges. So it will be better to count the number of st-edges at the input stage.

2. Count length-2 paths: for each vertex $v \neq s, t$, if there are both edges (s, v, c) and (v, t, c) for some color c, add solution by one and mark v not-alive.

3. count length-3 paths: for each pair u, v of alive vertices, if there are edges (s, u, c), (u, v, c), (v, t, c) for some color c, make the pair (u, v) an edge of another graph H. H is simple and undirected. Find a maximum cardinality matching of H.

4. Sum the total of the three steps as the solution.

Note that, after the first step, redundant edges are useless, i.e., two edges with the same endpoints and the same color act as one. We need a data structure to store the edges with different colors. Simply using c matrices is not good because of too much memory. One possible method is to encode the edges of the same endpoints by one integer. Since the color number is from 0 to 9, we may do this as follows:

```
scanf(``%d,%d,%d'',&u,&v,&c);
gr[u][v] |= (1<<c);
```

To test if two edges have a same color, we may use the bitwise-and operation.