514.自由之路

题目大意

给定一个环形表盘 $ring$，你需要通过进行一系列操作，来让表盘按顺序输出字符串 key 中的每一个字符。
你可以做的操作有两种：

1. 把指针移向当前位置的顺时针方向或逆时针方向的相邻位置
2. 让表盘输出当前指针指向的字符

已知当前指针指向 $ring[0]$，问至少需要做几次操作，能够让表盘输出 key

题目分析

观察到，如果需要完整输出字符串 key，需要固定地做 $key.length$ 次“按下”操作。因此我们只需计算“旋转”操作的最小次数。
对于每一次旋转，我们只关心当前的位置在哪里，然后计算当前位置与每一个下一个目标字符的位置的差值，由此即可进行递推。由于表盘为环形，我们可以写出一个计算两个位置之间“距离”的函数：

```
inline int dist(int x, int y) {
    if (x > y) swap(x, y);
    return min(y - x, abs(x + n - y));
}
```

我们使用一个数组 `vector<int> cand` 来记录表盘上字符为 $key[i]$ 的所有位置，并使用数组 `dp[i][now]` 来记录在转出 $key[i]$，此时转到表盘第 now 位时，所用的最小操作步数。从而可以写出状态转移方程：

$$dp_{now\in cand[i]}[i][now]=\min_{last\in cand[i-1]}(dp[i][now],dp[i-1][last]+dist(now,last))$$

我们先枚举 i，再枚举每一位 now，最后枚举每一位 $last$，时间复杂度为 $O(n^3)$，但是实际上常数非常小，大致趋近 $O(Const*n)$，完整代码如下：

```
int n, m, ans = 10000000;
int dp[105][105];
vector<int> cand[105];
inline int dist(int x, int y) {
```

```
    if (x > y) swap(x, y);
    return min(y - x, abs(x + n - y));
    int findRotateSteps(string ring, string key) {
        n = ring.length(), m = key.length();
        for (int i = 0;i < m;i++)
            for (int j = 0;j < n;j++)
                if (key[i] == ring[j]) cand[i].push_back(j), dp[i][j] = ans;
        for (auto ele : cand[0]) dp[0][ele] = dist(0, ele);
        for (int i = 1;i < m;i++)
            for (auto now : cand[i])
                for (auto last : cand[i - 1])
                    dp[i][now] = min(dp[i][now], dp[i - 1][last] + dist(now, last));
        for (auto ele : cand[m - 1]) ans = min(ans, dp[m - 1][ele]);
        ans += m;
        return ans;
    }
}
```