Code-

```
class Solution {
public:
    int f(int i, int j1, int j2, int n, int m, vector<vector<int>>& grid,
vector<vector<vector<int>>>& dp) {
        if (j1 < 0 || j2 < 0 || j1 >= m || j2 >= m) return -1e9;
        if (i == n-1) {
            if (j1 == j2) return grid[i][j1];
            else return grid[i][j1] + grid[i][j2];
        }
        if (dp[i][j1][j2] != -1) return dp[i][j1][j2];
        int maxi = -1e9;
        for (int dj1=-1; dj1 <= 1; dj1++) {
            for (int dj2=-1; dj2 <= 1; dj2++) {
                int value = 0;
                if (j1 == j2) value = grid[i][j1];
                else value = grid[i][j1] + grid[i][j2];
                value += f(i+1, j1+dj1, j2+dj2, n, m, grid, dp);
                maxi = max(maxi, value);
            }
        }
        return dp[i][j1][j2] = maxi;
    }
    int cherryPickup(vector<vector<int>>& grid) {
        int n = grid.size();
        int m = grid[0].size();
        vector<vector<vector<int>>> dp(n, vector<vector<int>>(m, vector<int>(m, -
1)));
        return f(0, 0, m-1, n, m, grid, dp);
    }
};
```

Time complexity -O(n^2)

Space complexity -O(n).