/\*

Given an array of non-negative integers, you are initially positioned at the first index of the array.

Each element in the array represents your maximum jump length at that position.

Determine if you are able to reach the last index.

For example:

A = [2,3,1,1,4], return true.

A = [3,2,1,0,4], return false.

方法一：贪心算法，就是找最远可以到达的位置

方法二：贪心算法，不用vector

\*/

class Solution {

public:

bool canJump(vector<int>& nums)

{

//way-1

/\*

if(nums.size()<2)

return true;

vector<int> steps(nums.size(),0);

int flag=0;

for(int i=0;i<nums.size()-1;i++)

{

if(steps[i]==0 && i!=0)

continue;

if(i+nums[i]<=flag)

continue;

else if(i+nums[i]>=nums.size()-1)

return true;

else

{

for(int j=flag+1;j<=i+nums[i];j++)

steps[j]=steps[i]+1;

}

flag=i+nums[i];//最远可以到达的区域

}

return false;

\*/

//way-2

int maxsize=0;

for(int i=0;i<nums.size()-1 && i<=maxsize;i++)

maxsize=max(maxsize,i+nums[i]);

if(maxsize>=nums.size()-1)

return true;

return false;

}

};