# Two sum

## 题目：

给定一个array数组存放若干个整数，给定一个目标数target，返回数组中的两个索引i，j，使得array[i] + array[j] == target

example:

Given nums = [2, 7, 11, 15], target = 9,

Because nums[0] + nums[1] = 2 + 7 = 9,

return [0, 1].

## 思路：

1 直接用两个for暴力求解，速度较慢。

2 对于array中每一个数a，总会存在另一个数b，使得a+b=target。//b不一定也存在数组array中  生成一个map，上述的a就是一个key，如果在数组array中存在b，那么b的索引就是value。  
 遍历数组array。 如果map[a]不存在，表示b目前不存在array中。对于b来说a已经存在数组中，则记下map[b] = a.index 如果map[a]存在,表示b已经在array中,返回结果a.index,map[a];

## 代码：

### 暴力求解

#include<iostream>

#include<vector>

using namespace std;

class Solution

{

public:

vector<int> twoSum(vector<int>& nums, int target)

{

vector<int> re;

int sz = nums.size();

for(int i=0;i<sz;++i)

{

for(int j=i+1;j<sz;++j)

{

if(nums[i]+nums[j]==target)

{

re.push\_back(i);

re.push\_back(j);

}

}

}

return re;

}

};

int main()

{

int a[]={11,7,2,5,9,0};

vector<int> n(a,a+6);

vector<int> b;

int tar=9;

Solution sol;

b=sol.twoSum(n,tar);

for(int i=0; i<b.size(); i++)

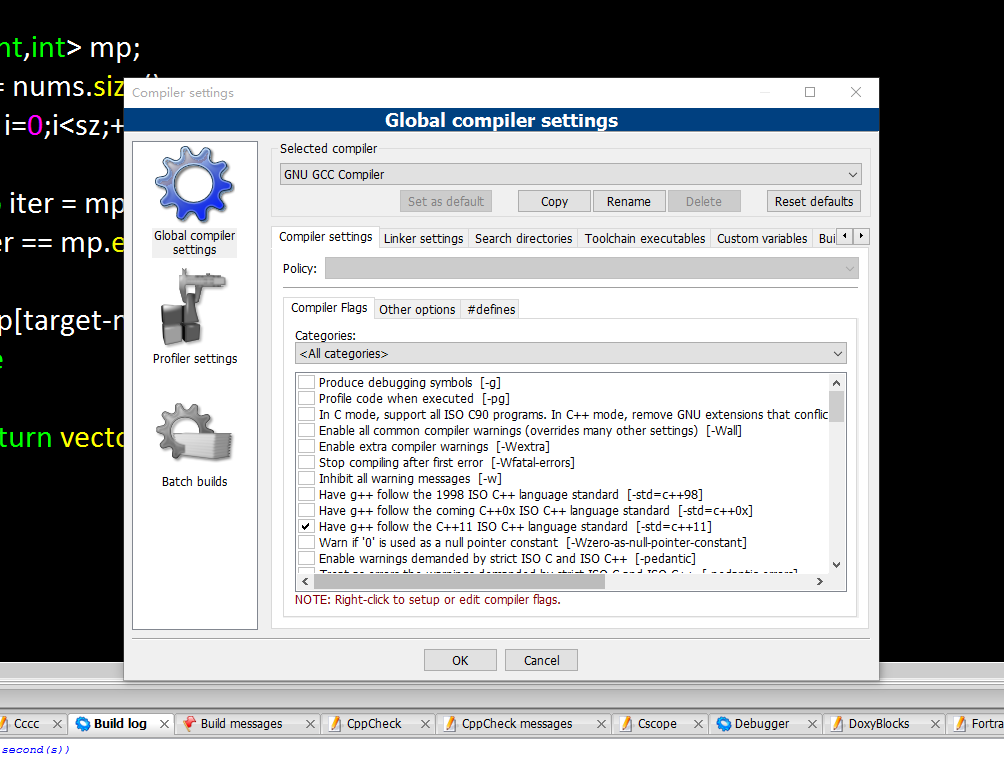
{

cout<<b[i]<<endl;

}

}

### 使用map



C++11标准支持，否则auto处会报错

代码：

#include<iostream>

#include<vector>

#include<map>

using namespace std;

class Solution {

public:

vector<int> twoSum(vector<int>& nums, int target)

{

map<int,int> mp;

int sz = nums.size();

for(int i=0;i<sz;++i)

{

auto iter = mp.find(nums[i]);

if(iter == mp.end())

{

mp[target-nums[i]] = i;

}else

{

return vector<int>{mp[nums[i]],i};

}

}

}

};

int main()

{

int a[]={11,7,2,5,9,0};

vector<int> n(a,a+6);

vector<int> b;

int tar=9;

Solution sol;

b=sol.twoSum(n,tar);

for(int i=0; i<b.size(); i++)

{

cout<<b[i]<<endl;

}

}