```
/*
给你一个整数数组 arr，请你帮忙统计数组中每个数的出现次数。

如果每个数的出现次数都是独一无二的，就返回 true；否则返回 false。



示例 1：

输入：arr = [1,2,2,1,1,3]
输出：true
解释：在该数组中，1 出现了 3 次，2 出现了 2 次，3 只出现了 1 次。没有两个数的出现次数相同。
示例 2：

输入：arr = [1,2]
输出：false
示例 3：

输入：arr = [-3,0,1,-3,1,1,1,-3,10,0]
输出：true


提示：

1 <= arr.length <= 1000
-1000 <= arr[i] <= 1000

来源：力扣（LeetCode）
链接：https://leetcode-cn.com/problems/unique-number-of-occurrences
著作权归领扣网络所有。商业转载请联系官方授权，非商业转载请注明出处。
*/
```

分析:

- 首先利用map数据结构完成出现次数统计;
- 然后利用set对出现次数进行去重;
- 比较map和set对象的长度是否相等.

方法一:C++_map_set

```
class Solution
{
    public:
        bool uniqueOccurrences(vector<int>& arr)
        {
            map<int,int> mii            ;
            set<int>     si             ;
            int          i      = 0     ;


            for(i=0;i<arr.size();i++)
```

```
            {
                if(mii.count(arr[i]))
                {
                    mii[arr[i]]++;
                }
                else
                {
                    mii[arr[i]] = 1;
                }
            }
            /*迭代器遍历*/
            map<int,int>::iterator iter;
            iter = mii.begin();
            while(iter != mii.end())
            {
                si.insert(iter->second);
                iter++;
            }
            return ( mii.size() == si.size() );
        }
};
/*
执行结果：
通过
显示详情
执行用时 :4 ms, 在所有 cpp 提交中击败了88.86%的用户
内存消耗 :8.8 MB, 在所有 cpp 提交中击败了100.00%的用户
*/
```

AlimyBreak
2019.10.21