```
/*
给出三个均为 严格递增排列 的整数数组 arr1，arr2 和 arr3。

返回一个由 仅 在这三个数组中 同时出现 的整数所构成的有序数组。



示例：

输入：arr1 = [1,2,3,4,5], arr2 = [1,2,5,7,9], arr3 = [1,3,4,5,8]
输出：[1,5]
解释：只有 1 和 5 同时在这三个数组中出现.



提示：

    1 <= arr1.length, arr2.length, arr3.length <= 1000
    1 <= arr1[i], arr2[i], arr3[i] <= 2000

来源：力扣（LeetCode）
链接：https://leetcode-cn.com/problems/intersection-of-three-sorted-arrays
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。
*/
```

- 遍历第一个数组,分别在另外一个数组中寻找是否有一样的，若有在另外两个数组中都有，就将当前值压入返回值数据结构;
- 可以使用map或set来简化查找运算.

方法一:C++_遍历+Map

```
class Solution
{
    public:
        vector<int> arraysIntersection( vector<int>& arr1,
                                        vector<int>& arr2,
                                        vector<int>& arr3
                                      )
        {

            vector<int> ret_val         ;
            int         i       =   0   ;
            map<int,int> mii2;
            map<int,int> mii3;

            for( i = 0 ; i < arr2.size();i++)
            {
                mii2[arr2[i]] = 1;
            }
            for( i = 0 ; i < arr3.size();i++)
            {
                mii3[arr3[i]] = 1;
            }
```

```
23
24              for(i = 0 ; i < arr1.size(); i++)
25              {
26                  if(   ( mii2.count(arr1[i]) > 0 )
27                     && ( mii3.count(arr1[i]) > 0 )
28                  )
29                  {
30                      ret_val.push_back(arr1[i]);
31                  }
32              }
33              return ret_val;
34          }
35  };
36  /*
37  执行结果:
38  通过
39  显示详情
40  执行用时 :28 ms, 在所有 C++ 提交中击败了100.00% 的用户
41  内存消耗 :11.8 MB, 在所有 C++ 提交中击败了100.00%的用户
42  */
```

AlimyBreak
2019.10.12