提交方式：

Canvas 上提交

提交内容：

一个压缩包，命名方式：姓拼音_名拼音_学号.zip/rar，如：zhou_zhiming_2020000112.zip（不要用中文）

内含若干个文件夹，每个文件夹对应一道题的项目（仅保留.pro 和其他源文件；文件夹名也不要有中文）

(.cpp / .h 等)

评分标准：

代码正确性和完整性 90%

代码风格 10% （代码看着很乱的酌情扣 0-10 分。唯一的要求：代码需按层次缩进、对齐）

逾期惩罚：

逾期提交的，成绩 *= 0.8 （无论逾期多久）

以下题目均要求：基于 Set 或 Map 完成，不能使用 set 或 map

1. 保序去重
   (1) Console 输入：
      ① 一个 Vector<int>，以{1,3,1,2}的形式
   (2) Console 输出：
      ① 去重后的 Vector，以{1,3,2}的形式（保持原出现先后顺序）

   提示：基于 Set 的 contrains 函数判断是否存在重复

2. 数独
   (1) Console 输入：
      ① 一个 Grid<int>，以{{1,2,3},{2,3,1},{3,2,1}}的形式
      ② 不保证元素数值在 1-9 之间，不保证 Grid 大小为 9x9
   (2) Console 输出：
      ① 输入的 Grid 是否为一个满足要求的数独

   提示：通过 Set 判断每行、每列、每个单元格是否由 1-9 组成

3. 加密与解密
   (1) 一种经典的加密方法是 26 个字母打乱随机映射
   假设映射表为：
   ABCDEFGHIJKLMNOPQRSTUVWXYZ
   IXECGQPSWFOAUYDBRJTKZMHLVN
   则 programming 被加密为 bjdpjiuuwyp

   (2) 任务一：已知映射表的加解密
      ① Console 输入：
         1) 前两行，每行 26 个字母，表示加密映射表
         2) 第三行，一个原单词
         3) 第四行，一个加密后的单词
      ② Console 输出：
         1) 第一行，第三行单词对应的加密后的单词
         2) 第二行，第四行单词对应的原单词

   提示：
   建立 Map<char, char> orig_to_encode; 原文为 key，密文为 value。用于加密
   建立 Map<char, char> encode_to_orig; 密文为 key，原文为 value。用于解密

   (3) 任务二：尝试破译（感兴趣的同学）
      ① 输入：
         1) 一个用这种方法加密后的单词
      ② 输出：
         1) 在给定的单词表中（文件 words.txt 下载 words.txt 中）的所有可能的原单词
      ③ 注：只知道加密后的单词，不知道加密映射表。即，任务是尝试进行解密

   提示：因字母被打乱，故仅需考虑单词内字母排布格式是否一致

4. 成绩单排序

(1) Console 输入：

① 第一行，一个整数 n，表示学生总数

② 接下来 n 行，每行一个姓名和一个成绩，用空格隔开，可以假设姓名内部没有空格

(2) Console 输出：

① 分数从高到低排序后的成绩单

② n 行，每行一个姓名和一个成绩，用空格隔开

提示：可使用 Map<int, Set<string>>。成绩为 key，得对应成绩的学生的名字的集合为 value

5. 绩点计算器

(1) 给定：

① 一个某同学的成绩单：grade.txt

1) 每行一个字符串、两个整数，用空格隔开，代表一门课程的名称、学分与成绩

2) 如：chengxusheji 4 95，表示 chengxusheji 这门课，4 学分，成绩为 95

② 一个绩点表： gpa.txt

1) 每行两个整数一个浮点数，用空格隔开，代表分数区间和对应绩点

2) 如：90 100 4.0，表示 90 ~ 100 分的绩点为 4.0

(2) 输出：

① 分行列出学生的绩点分布，格式如下：

1) 绩点 -- 课程名字(学分, 成绩) 课程名字(学分, 成绩) ... 课程名字(学分, 成绩)

2) 如：4.0 -- chengxusheji(4, 95) python(3, 80)

② 最后给出按学分加权的平均绩点

提示：

先读入绩点文件，建立 Map<int, double> chengji_to_jidian；成绩为 key，绩点为 value

形如：{...,{85, 3.7}, ..., {89, 3.7}, {90, 4.0}, {91, 4.0}, ...{100, 4.0}}

再读入成绩文件，建立 Map<double, Vector<Course>> jidian_to_courses; 绩点为 key，课程列表为 value

最后按要求输出 jidian_to_courses，并计算平均绩点