1.从小明家所在公交站出发有 n 路公交到公司，现给出每路公交的停站数(不包括起点和终点)，及每次停的时间(一路车在每个站停的时间相同)和发车的间隔，先假定每辆车同时在相对时间 0 分开始发车，且所有车在相邻两个站之间的耗时相同,都为 5 分钟。给定小明起床的相对时间(相对 0 的分钟数)，请计算他最早到达公司的相对时间。
给定每路车的停站数 stops,停站时间 period,发车间隔 interval 及公交路数 n，出发时间 s。请返回最早到达时间。保证公交路数小于等于 500，停站数小于等于 50。

```
import java.util.*;

public class TakeBuses {
  public int chooseLine(int[] stops, int[] period, int[] interval, int n, int s) {
    // write code here
    int min = Integer.MAX_VALUE;
    for(int i=0;i<n;i++){
      int missTime = s%interval[i];
      int waitCost = missTime==0?0:interval[i]-missTime;
      min = Math.min(min,waitCost+(stops[i]+1)*5+stops[i]*period[i]);
    }
    return min+s;
  }
}
```

2. 请你实现一个简单的字符串替换函数。原串中需要替换的占位符为"%s",请按照参数列表的顺序一一替换占位符。若参数列表的字符数大于占位符个数。则将剩下的参数字符添加到字符串的结尾。
给定一个字符串 A，同时给定它的长度 n 及参数字符数组 arg，请返回替换后的字符串。保证参数个数大于等于占位符个数。保证原串由大小写英文字母组成，同时长度小于等于 500。

测试样例："A%sC%sE",7,['B','D','F']　　返回："ABCDEF"

```
public String formatString(String A, int n, char[] arg, int m) {
      int i = 0;
    while (A.indexOf("%s") >= 0) {
      A = A.replaceFirst("%s", String.valueOf(arg[i]));
      i++;
    }
    while (i < m) {
      A += arg[i];
      i++;
    }
    return A;
}
```

3. 现在有一个字符串列表，和一个关键词列表，请设计一个高效算法，检测出含关键字列表中关键字(一个或多个)的字符串。
给定字符串数组 A 及它的大小 n 以及关键词数组 key 及它的大小 m，请返回一个排好序的含关键词的字符串序号的列表。保证所有字符串长度小于等于 100，关键词个数小于等于 100，字符串个数小于等于 200。保证所有字符串全部由小写英文字符组成。若不存在含关键字的

字符串，请返回一个只含-1 的数组。
测试样例：
["nowcoder","hello","now"],3,["coder",now],2
返回：[0,2]

```
import java.util.*;

public class KeywordDetect {
   public int[] containKeyword(String[] A, int n, String[] keys, int m) {
      List<Integer> list = new ArrayList<Integer>();
      for(int i = 0;i<n;i++){
         for(int j = 0;j<m;j++){
            if(A[i].indexOf(keys[j])>-1){
               list.add(i);
               break;
            }
         }
      }
      if(list.isEmpty())
         return new int[]{-1};
      int[] r = new int[list.size()];
      for(int i = 0;i<list.size();i++)
         r[i] = list.get(i);
      return r;

   }
}
```

4.
血型遗传对照表如下：

父母血型	子女会出现的血型	子女不会出现的血型
O 与 O	O	A,B,AB
A 与 O	A,O	B,AB
A 与 A	A,O	B,AB
A 与 B	A,B,AB,O	——
A 与 AB	A,B,AB	O
B 与 O	B,O	A,AB

B 与 B	B,O	A,AB
B 与 AB	A,B,AB	O
AB 与 O	A,B	O,AB
AB 与 AB	A,B,AB	O

请实现一个程序，输入父母血型，判断孩子可能的血型。
给定两个字符串 father 和 mother，代表父母的血型,请返回一个字符串数组，代表孩子的可能血型(按照字典序排列)。
测试样例：
”A”,”A”
返回：[”A”,“O”]

```
class ChkBloodType {
public:
    map<string,vector<string>> rules {
        {"OO", {"O"}},
        {"AO", {"A","O"}},
        {"AA", {"A", "O"}},
        {"AB", {"A","AB","B","O"}},
        {"AAB", {"A","AB","B"}},
        {"BO", {"B","O"}},
        {"BB", {"B","O"}},
        {"BAB", {"A","AB","B"}},
        {"ABO", {"A","B"}},
        {"ABAB", {"A","AB","B"}}
    };
    vector<string> chkBlood(string father, string mother) {
        vector<string> result;
        if(rules.find(father + mother) != rules.end()){
            result = rules[father + mother];
        } else if (rules.find(mother + father) != rules.end()){
            result = rules[mother + father];
        }

        return result;
    }
};
```