Difficulty: Easy Points: 20

Problem: GEM-STONES

John has collected various rocks. Each rock has various minerals embeded in it. Each type of mineral is designated by a lowercase letter in the range . There may be multiple occurrences of a mineral in a rock. A mineral is called a *gemstone* if it occurs at least once in each of the rocks in John's collection.

Given a list of minerals embedded in each of John's rocks, display the number of types of gemstones he has in his collection.

Input Format

The first line consists of an integer , the number of rocks.
Each of the next lines contains a string where each letter represents an occurence of a mineral in the current rock.

Constraints

Each composition consists of only lower-case Latin letters ('a'-'z').
length of each composition

Output Format

Print the number of types of gemstones in John's collection. If there are none, print .

Sample Input

```
3
abcdde
baccd
eeabg
```

Sample Output

```
2
```

Explanation

Only and are gemstones. They are the only types that occur in every rock.

Solution:

```
int contains(string str, char temp)
  {
    int flag = 0;
    int length = str.length();
    for(int i=0; i<length; i++)
     {
       if(str[i]==temp)
          {flag=1;}
     }
    if(flag==1)
       return 1;
    return 0;
  }

int main() {
  /* Enter your code here. Read input from STDIN. Print output to STDOUT */
  int cases;
  string duplicate="";
  int length, index = 0, counter = 0;
  cin>>cases;
  string array[cases];
  for(int i=0; i<cases; i++)
    {
      cin>>array[i];
      int temp = array[i].length();
      if(i==0)
        length = temp;
      else if(i!=0 && temp<length)
        {
          length = temp;
          index = i;
        }
  }

  int result = 0;
  string str = array[index];
```

```
  length = str.length();
  for(int i=0; i<length; i++)
    {

      char temp = str[i];
      if(!contains(duplicate, temp))
      {
        counter=0;
        for(int j=0; j<cases; j++)
          {
            if(j!=index)
              {
                if(!contains(array[j], temp))
                  { counter+=0; }
                else
                  {counter +=1;}
              }
          }
        if(counter == cases-1)
          {result+=1;
           duplicate+=temp;
          }
      }
  }
  cout<<result;
  return 0;
}
```

Elegant Solution

```
public class Solution
        {
            public static void main(String[] args)
              {

                    Scanner sc = new Scanner(System.in);
                    int cases = sc.nextInt();
                    Set <Character> set = pick(sc.next()); //input the first string in the set

                    for(int i=0; i<cases-1; i++)
                      {
                        set.retainAll(pick(sc.next()));
                      }
                    System.out.println(set.size());
                }
            public static Set<Character> pick(String str)
                {
                     Set<Character> set = new HashSet<Character> (26);
                     for(char c: str.toCharArray())
                       {
                          set.add(Character.valueOf(c));
                       }
                     return set;
                }
        }
```

`'Anshul AGgarwal