Chapter 21. Sets and Maps

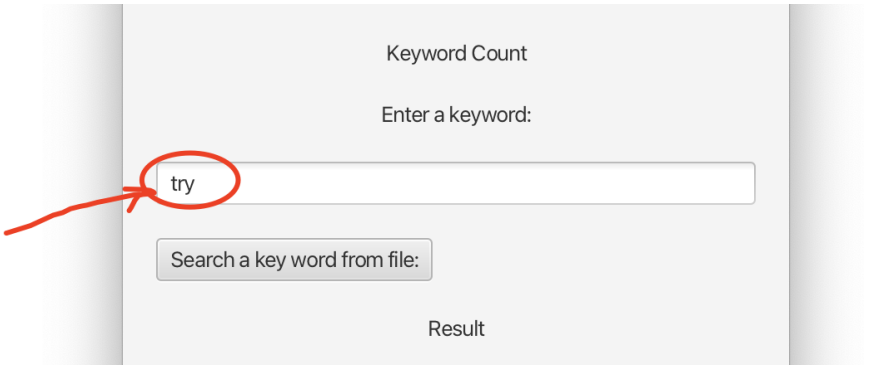
Objectives:

To use sets to develop a program that counts the keywords in a Java source file (§21.4).
To use maps to develop a program that counts the occurrence of the words in a text (§21.6).

Problem A

Create a GUI program that calculates the number of times a user-entered keyword appears in a Java code file.

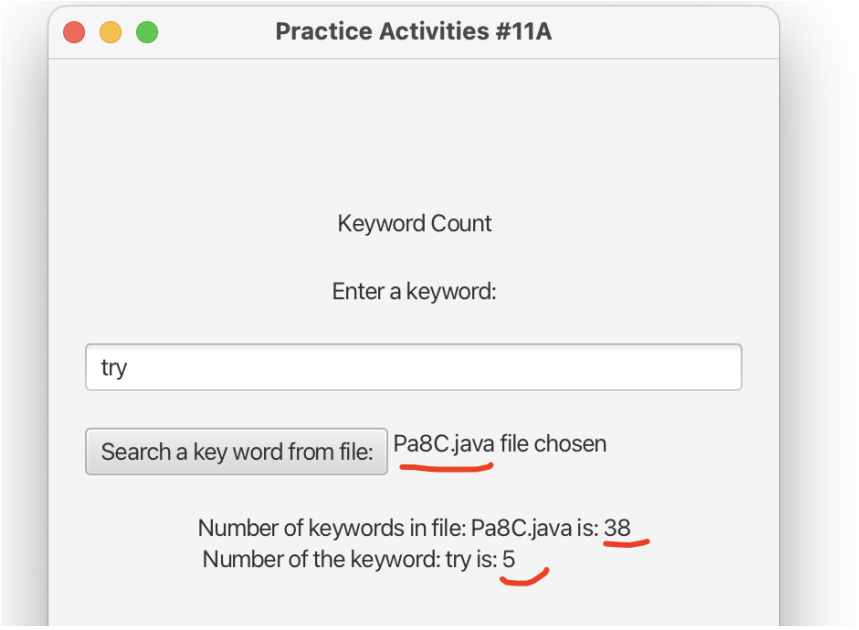
1: User inputs a keyword



2: Choose a file(only *.java) with a button click (use FileChooser) and see the result.

Result includes:

- the chosen file name is displayed
- total amount of keywords in the java file
- the number of occurrences of the user's keyword in the code.



List of KeyWords:

```
String[] keywordString = {"abstract", "assert", "boolean",
        "break", "byte", "case", "catch", "char", "class", "const",
        "continue", "default", "do", "double", "else", "enum",
        "extends", "for", "final", "finally", "float", "goto",
        "if", "implements", "import", "instanceof", "int",
        "interface", "long", "native", "new", "package", "private",
        "protected", "public", "return", "short", "static",
        "strictfp", "super", "switch", "synchronized", "this",
        "throw", "throws", "transient", "try", "void", "volatile",
        "while", "true", "false", "null"};
```

Use a Set to store keywords:

```
Set<String> keywordSet =
      new HashSet<>(Arrays.asList(keywordString));
```

Problem B

Create a method `countCharacter()`, that accepts any String from the user and converts it into a character array and counts the number of times each character is present in the array.

Add how many times each character is present to a Hashmap with the character as *key* and the repetitions count as *value*.

Optut both array and map.

`Ex:` input: Apple

Output: char[] {'A','P',P','L','E'}

HashMap {'A':1, 'P':2, 'L' :1,'E':1}

Problem C

In this problem, you will work with the methods of the `Map` interface to support adding new phone numbers into a phonebook.

The numbers are represented by a class named `PhoneNumber` consisting of only two fields: `PhoneNumberType type;` and `String number;` The `PhoneNumberType` enum determines the type of the number.

The phonebook is a class based on a map where keys are names and values are lists of phone numbers because each person can have multiple phone numbers.

To solve this problem, you need to implement two methods of the `PhoneBook` class.

- The `addNewPhoneNumbers` method should add given phone numbers to the list of the numbers for a specified person by the name. If the name is not yet in the phonebook, then it must appear in it with the given numbers.

- The `printPhoneBook` should print all numbers with their types for each name in the phonebook.

Edit the code:

```
class PhoneBook {
   private final Map<String, Collection<PhoneNumber>> nameToPhoneNumbersMap = new
HashMap<>();

   public void addNewPhoneNumbers(String name, Collection<PhoneNumber> numbers) {
       // write your code here
   }
   public void printPhoneBook() {
       // write your code here
}
enum PhoneNumberType {
   MOBILE, HOME, WORK,
}
class PhoneNumber {
   private PhoneNumberType type;
   private String number;

   public PhoneNumber(PhoneNumberType type, String number) {
       this.type = type;
       this.number = number;
   }
   public PhoneNumberType getType() {
       return type;
   }
   public String getNumber() {
       return number;
   }
   //test
```

```
    public static void main(String[] args) {
        PhoneBook phoneBook = new PhoneBook();

        List<PhoneNumber> claraPhoneNumbers = new ArrayList<>();
        claraPhoneNumbers.add(new PhoneNumber(PhoneNumberType.HOME, "723324324"));
        phoneBook.addNewPhoneNumbers("Clara", claraPhoneNumbers);

        List<PhoneNumber> kevinPhoneNumbers = new ArrayList<>();
        kevinPhoneNumbers.add(new PhoneNumber(PhoneNumberType.WORK, "1231"));
        phoneBook.addNewPhoneNumbers("Kevin", kevinPhoneNumbers);

        phoneBook.addNewPhoneNumbers("Clara", List.of(new
PhoneNumber(PhoneNumberType.MOBILE, "23424279")));
        phoneBook.addNewPhoneNumbers("Paul", List.of(new
PhoneNumber(PhoneNumberType.WORK, "56756335")));

        phoneBook.printPhoneBook();
    }
}
```

After executing this code, if everything is OK, your program must print:

```
Kevin
WORK: 1231
Clara
HOME: 723324324
MOBILE: 23424279
Paul
WORK: 56756335
------
```