Problem solving:
Extract parts of a phone number

String processing with indexOf(), and substring()

General solution vs Hardcoding
Problem solving and
General solution vs hardcoded or too specific solutions

Write a program that asks the user to enter a phone number (as a string in the format xxx-xxx-xxxx) and extracts the area code and the last 4 digit of the number.

Enter your phone number: 817-123-8888
Area code: 817
Last 4 digits: 8888

Problem solving:
Step 1: figure out input, output, needed data, the type for each of them and the variable name for each of them.
Input data: phone number
    stored as what type (and what variable name)? String phone

What data does the program need to compute?
    area code (type? Choose String) -> String area
    last 4 digits (type? Choose String) -> String rest
    (here we chose type String b.c. input, phone number, is a String)

Do I need to compute intermediate data? Maybe
Problem solving: Step 2: Computing data

To compute the area code
- I need a small piece of another string. Is there any method that does that? Yes: substring
- (for a hw, at this step you would review the substring method if you were not familiar with it)
- Write the indexes for the sample phone number
- How would I get 817 for the example above ("817-123-8888")?
  - V1: "817-123-8888".substring(0, 3) Hardcoded, will not work if user gives 662-....
  - V2: phone.substring(0, 3) will get the first 3 symbols from the user’s string, but it still has some hardcoding. What if the user has some spaces? E.g. 8 17-12 3 - 8888?
- V3: A more general/flexible solution: Copy everything from 0 up to the first ".". What do I need to make it work? The index of the first ".". Any String method that gives that? Yes. indexOf. =>
  ```
  int idx = phone.indexOf("-");
  String area = phone.substring(0, idx); // symbol pt index idx will not be included.
  ```
  (better to do the work in 2 steps: first compute idx, second, use idx to compute area)

To compute the last 4 digits
- General solution: copy from the phone string everything after the 2nd "-".
  - Can I copy from a string everything from a specific index to the end? Yes. E.g. s.substring(5).
  - Can I find the index of the 2nd "-"? Let’s say I do not know any such method. Can I reduce this problem to something I know? I do know how to find the index of the first "-". So now I need to solve this smaller problem: Given a string with two "-" in it, can I extract part of it that has only the 2nd "-" and what comes after it? Yes: copy everything after the first "-" to the end=>
  ```
  String temp = phone.substring(1+idx); // where idx = phone.indexOf("-")
  now temp should have "123-8888"
  ```
  - Now I need to extract from temp what comes after the "-" in temp:
    ```
    String last4 = temp.substring(1+ temp.indexOf("-")); // last4 should have "8888"
    ```
Put all these together and write the code. (Ideally, write code as you develop this.)
Use the Canvas viewer to see the data.
Scanner in = new Scanner(System.in);
System.out.print("Enter the phone number: ");
String phone = in.nextLine(); // chose this because it allows for spaces as well
   // say phone = "817-123-8888"
String area, last4, temp;
int idx = phone.indexOf("-");
area = phone.substring(0,idx); //symbol at index idx will not be included
   // area should be "817"
temp = phone.substring(1+idx); // I do not want to include the first -
   // temp should have: "123-8888"
last4 = temp.substring(1+ temp.indexOf("-"))); //note that here we use temp, not phone
   // temp.indexOf("-")) should give 3
   // temp.substring(1+3) should return "8888"
   // last4 should be "8888"
System.out.println("Area code: " + area);
System.out.println("The last 4 digits are: " + last4);

Sample run 1 (with given data)
Enter the phone number: 817-123-8888
Area code: 817
The last 4 digits are: 8888

Sample run 1 (with weird/unexpected data)
Enter the phone number: 8 17 - 123 - 8 8 8 8
Area code: 8 17
The last 4 digits are: 8 8 8 8

NOTE: this solution does not only work with spaces, but also with other types of phone numbers and area
codes: e.g. 67-6886-76876 or 4532675-743-4763.
Moreover, you can do the same processing with other separators. E.g. "." in domains such as uta.edu or
"/" in webpage addresses e.g. http://vlm1.uta.edu/~alex/courses/1310/lectures/_lectures.html
⇒ Solution to a type of problem, not an instance. If you use a variable e.g. String sep = ":-"; then you will only
need to modify that to String sep = ":"; and it works for ":"
Using the Java API Library

- Note that there is a Java method that gives the index of the LAST occurrence of a substring in a string.
- Look it up, read it’s signature, rewrite your problem solution to use it.
- So why do all that complicated processing and not use this method?
  - It is good to take advantage of methods that are already available, but by not using it, we forced ourselves to “work some more on the data”, and learnt how to approach it and were able to solve the problem with what we already know.
  - There may not always be a method ready in the API for you
  - The processing that we did will still be needed if I have to extract more than exactly 2 pieces of the substring. What if there is any number of “-” in the string and I have to extract every substring in between them?