PS 2231 2025 Spring
Lab3: Review and n-dim Arrays

Instructor: Dr. Y. Tiffany Tang[1] and Dr. Pinata Winoto[2]
Songlin Shang[3], Cong Ye[4] and Zike Deng[5]

Instruction:

Self-check: In all labs in the future, you can download `check.py` in canvas, which will help you check the output format in labs, to make sure you don't have compile error or logic error before you submit your file. If you didn't finish installing the python environment/JDK installation, you can download the file from canvas named as tutorial.pdf. Please read and follow the tutorial, which teaches you how to install a python environment, for checking your code result. Also, the instructions of all questions also teach you how to use `check.py` to check your code.

Plagiarism: We encourage discussing all labs with your classmate, but **NEVER** copy other's code directly. We will use GradeScope[@] platform, to check the similarity probability, 90% similarity reported by GradeScope will be reviewed by TAs, and will report to your instructor and receive some penalty based on the event.

AI-Generate code issue: We encourage you to use AI tools, to help you compile all labs, **BUT NEVER** directly **copy and paste**. To detect that, we will use different tool to detect AI possibility include:

ChatGPT Code Detector:https://chatgpt.com/c/67bdddec-c728-8010-b1b6-abf3e56b3136

GPTSniffer: https://github.com/MDEGroup/GPTSniffer

Penalty will be given for high AI probability.

Grading: Each lab has the same points, and each test case has the same points. In check.py, you might check your code with given test cases in labs to avoid spelling errors and compile errors. We still have invisible test cases, which test for grading. This might test you to consider all possible positions. All test cases have the same points.

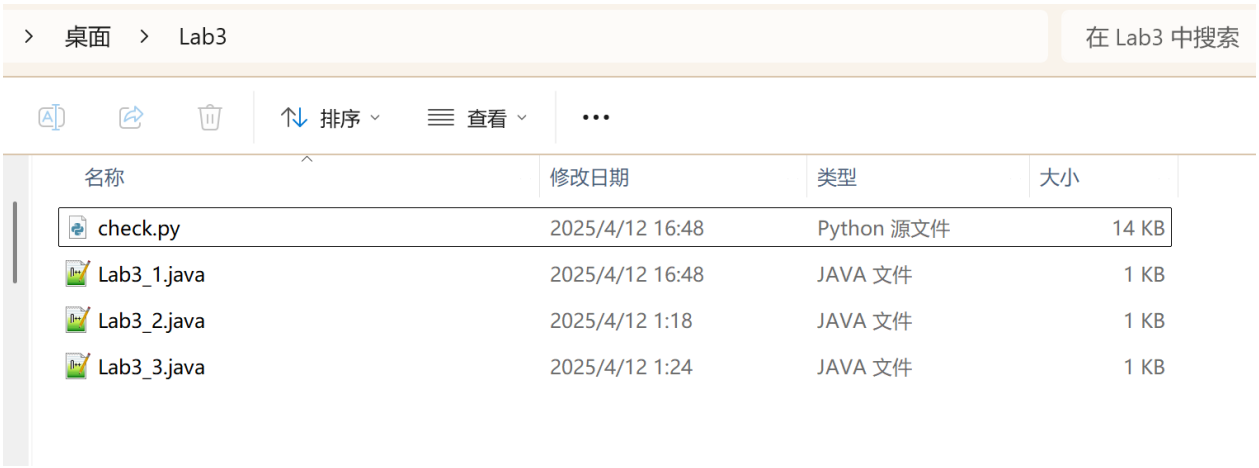
[1] Tiffany Ya Tang, Associate Professor, Department of Computer Science, Wenzhou-Kean University; Email: yatang@kean.edu
[2] Pinata Winoto, Assistant Professor, Department of Computer Science, Wenzhou-Kean University; Email: pwinoto@kean.edu
[3] Songlin Shang, Department of Computer Science, Wenzhou-Kean University; Email: shangs@kean.edu
[4] Cong Ye, Department of Computer Science, Wenzhou-Kean University; Email: yecon@kean.edu
[5] Zike Deng, Department of Computer Science, School of Computing and Data Science, University of Hong Kong; Email: baylordeng@connect.hku.hk

Get Started

Please download Lab3_1.java, Lab3_2.java, Lab3_3.java on canvas.

Before you start the lab, you can create a project on eclipse called Lab3. Also, we suggest you download the check.py in canvas and move it under the same folder with all labs.

Before you start coding, please direct you terminal to same folder using following code: `cd path (path is the way to the folder of check.py)`
here is an example:

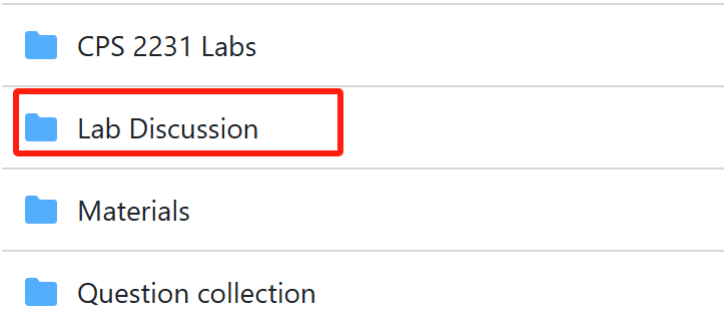


P.S.

Lab3 is optional and provides an opportunity to boost your score.
If your previous two problems are completely correct, you will earn the full **40 points**. However, if you choose to challenge **Lab3 (Lab2_3)**, your current score will be multiplied by **1.1**—but your final score will **not exceed 40 points**.

For example, if you scored 36 points on the first two labs and Lab3 is fully correct, your final score would be **36 × 1.1 = 39.6**.

Lab3 is not very difficult. Detailed instructions and hints are provided in the **Hint** sections. We may provide the idea of solving in Github. You'll have a bit of time to think things through before this material goes live. Keep an eye out — updates might be coming soon!



Just give it a try—it's a good opportunity to practice and improve!

Now you can start coding after all these preparations.

N.B.: **Please replace Li Hua with your Chinese name in Pinyin in following labs.**

Please don't use anything that hasn't been introduced or taught in CPS 1231 or 2231, otherwise there will be a penalty.

Lab 3_1

Write a method named **containsBothDigits** that takes an integer n and **returns** true if the number contains both digits 2 and 7, and false otherwise.

Example 1
Input: 1237
Output: After Li Hua's testing, 1237 contains both 2 and 7, which is true

Example 2
Input: 0.27
Output: Invalid Input.

Example 3
Input: 2
Output: After Li Hua's testing, 2 contains both 2 and 7, which is false

Example 4
Input: a
Output: Invalid Input.

Example 5
Input: -28272567
Output: After Li Hua's testing, -28272567 contains both 2 and 7, which is true

Self-Check:

When you finish coding, you can run following code: `python check.py --lab Lab3_1`

If everything good, you might see following content:

********* Compiling Java file: Lab3_1.java *********

☑ Compilation succeeded!

Running test case: Valid input 1237

********* Running Java program: Lab3_1 *********

☑ Execution succeeded!

☑ Output is correct: After Li Hua's testing, 1237 contains both 2 and 7, which is true

Running test case: Invalid input 0.27

********* Running Java program: Lab3_1 *********

☑ Execution succeeded!

☑ Output is correct: Invalid Input.

Running test case: Valid input 2

********* Running Java program: Lab3_1 *********

☑ Execution succeeded!

☑ Output is correct: After Li Hua's testing, 2 contains both 2 and 7, which is false

Running test case: Invalid input a

********* Running Java program: Lab3_1 *********

☑ Execution succeeded!

☑ Output is correct: Invalid Input.

Running test case: Valid input -28272567

********* Running Java program: Lab3_1 *********

☑ Execution succeeded!

☑ Output is correct: After Li Hua's testing, -28272567 contains both 2 and 7, which is true

☑ All checks passed for the specified lab!

Lab 3_2

Two strings are considered anagrams if they contain the same characters, but the characters may be arranged in a different order. Write a method called **areAnagrams** and returns true if two strings are anagrams of each other, otherwise false. Treat the input as **case-insensitive** and only read alphanumeric characters.

Hint:

While comparing str1 and str2, you can create a boolean array to track which characters in str2 have already been matched. This helps **handle duplicate characters** properly.

During the comparison process, once a character in str2 has been matched with a character in str1, it should not be matched again. The boolean array allows you to **skip already-used characters** in subsequent comparisons

Example 1
Input:
Astronomer
Moon starer
Output: After Li Hua's testing, astronomer and moonstarer are anagrams, which is true

Example 2
Input:
hello123
3he1llo2
Output: After Li Hua's testing, hello123 and 3he1llo2 are anagrams, which is true

Example 3
Input:
Java
Python
Output: After Li Hua's testing, java and python are anagrams, which is false

Example 4
Input:
GameMaster!
Master Game
Output: After Li Hua's testing, gamemaster and mastergame are anagrams, which is true

Example 5
Input:
!

_
Output: After Li Hua's testing, and are anagrams, which is true

Self-Check:

When you finish coding, you can run following code:

```
python check.py --lab Lab3_2
```

If everything good, you might see following content:
********* Compiling Java file: Lab3_2.java *********

☑ Compilation succeeded!

Running test case: Valid input: anagrams

********* Running Java program: Lab3_2 *********

☑ Execution succeeded!

☑ Output is correct: After Li Hua's testing, astronomer and moonstarer are anagrams, which is true

Running test case: Valid input: anagrams with numbers

********* Running Java program: Lab3_2 *********

☑ Execution succeeded!

☑ Output is correct: After Li Hua's testing, hello123 and 3he1llo2 are anagrams, which is true

Running test case: Valid input: not anagrams

********* Running Java program: Lab3_2 *********

☑ Execution succeeded!

☑ Output is correct: After Li Hua's testing, java and python are anagrams, which is false

Running test case: Valid input: anagrams with special characters

********* Running Java program: Lab3_2 *********

☑ Execution succeeded!

☑ Output is correct: After Li Hua's testing, gamemaster and mastergame are anagrams, which is true

Running test case: Valid input: special characters

********* Running Java program: Lab3_2 *********

☑ Execution succeeded!

☑ Output is correct: After Li Hua's testing, and are anagrams, which is true

☑ All checks passed for the specified lab!

Lab 3_3 Challenge Question (Optional)

Columnar String Encoding

Write a method called **encode** that takes a string s and an integer n as parameters and that can print a new string that scrambles the order of the characters from s in a particular way. The characters should be placed into a grid with n rows and a number of columns determined by the length of the string. The characters are placed into the grid **column by column**. After filling the grid, the characters should be concatenated **by row**.

Hint: You may assume that the string passed as a parameter is not empty and that the integer passed as a parameter is greater than or equal to 1 and less than the length of the string. The string might contain any characters, including spaces.

Example 1
Input:
four score and seven
4
Output:
Row 1: f r n e
Row 2: o s e d v
Row 3: u c e
Row 4: r o a s n
After Li Hua's testing, Final Encoded String is f rneosedvuc eroasn

Example 2
Input:
hello world!
3
Output:
Row 1: h l w l
Row 2: e o o d
Row 3: l r !
After Li Hua's testing, Final Encoded String is hlwleoodl r!

Example 3
Input:

Output:
Invalid Input.

Example 4
Input:
2231-25SP
-1
Output:
Invalid Input.

Example 5
Input:
abcdefghijklmnopqrstuvwxyz
3
Output:
Row 1: a d g j m p s v y
Row 2: b e h k n q t w z
Row 3: c f i l o r u x
After Li Hua's testing, Final Encoded String is adgjmpsvybehknqtwzcfilorux
Self-Check:

When you finish coding, you can run following code: `python check.py --lab Lab3_3`

If everything good, you might see following content:

********* Compiling Java file: Lab3_3.java *********

☑ Compilation succeeded!

Running test case: Valid input: sentence with 4 rows, spaces preserved

********* Running Java program: Lab3_3 *********

☑ Execution succeeded!

☑ Output is correct: Row 1: f   r n e
Row 2: o s e d v
Row 3: u c   e
Row 4: r o a s n
After Li Hua's testing, Final Encoded String is f rneosedvuc eroasn

Running test case: Valid input: sentence with 3 rows, spaces preserved

********* Running Java program: Lab3_3 *********

☑ Execution succeeded!

☑ Output is correct: Row 1: h l w l
Row 2: e o o d
Row 3: l   r !
After Li Hua's testing, Final Encoded String is hlwleoodl r!

Running test case: Invalid input: empty string

********* Running Java program: Lab3_3 *********

☑ Execution succeeded!

☑ Output is correct: Invalid Input.

Running test case: Invalid input: negative row count

********* Running Java program: Lab3_3 *********

☑ Execution succeeded!

☑ Output is correct: Invalid Input.

Running test case: Valid input: alphabet with 3 rows

********* Running Java program: Lab3_3 *********

☑ Execution succeeded!

☑ Output is correct: Row 1: a d g j m p s v y
Row 2: b e h k n q t w z
Row 3: c f i l o r u x
After Li Hua's testing, Final Encoded String is adgjmpsvybehknqtwzcfilorux

☑ All checks passed for the specified lab!

Last Step

Congratulations! You have finished all Labs in Lab2.

Now please run the following code in the end:

```
python check.py --uid 123456
```

(replace 123456 as you student number)

It will check all labs again and zip a file. **Since Lab 3_3 is an optional lab,** it will ask you to answer whether you have finish the lab, and you might see following content:

```
Have you completed Lab3_3? (y/n):
```

Input " y",then press " enter" if you finish Lab2_3, otherwise press n.

If everythings good, you might see following information:

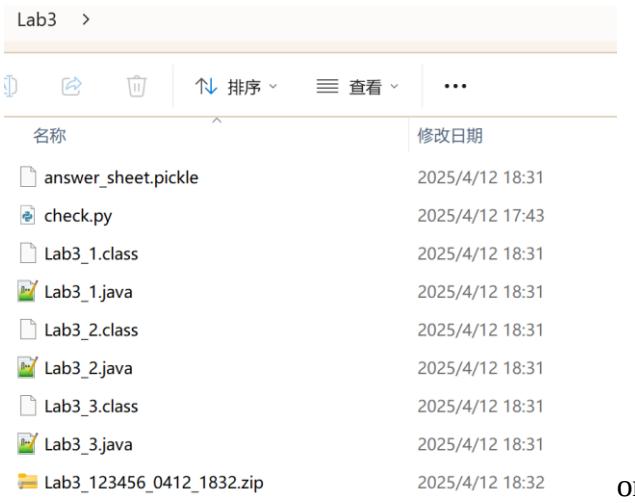
```
Your Student ID is 123456. Is this correct? (y/n):
```

Input " y",then press " enter", you will see following information.

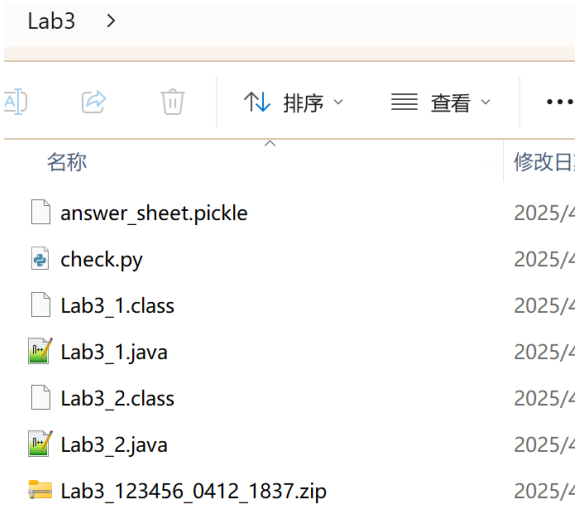
```
********* Zipping Files *********
Zipping Lab3_1.java... Done.
Zipping Lab3_2.java... Done.
Zipping answer_sheet.pickle... Done.
Zipping Lab3_3.java... Done.

📦 Successfully created submission package: Lab3_123456_0412_1832.zip
```

You will see there is a new file under the folder like:



or



Submit this .zip file to canvas!