COSC 1P03 Assignment 1

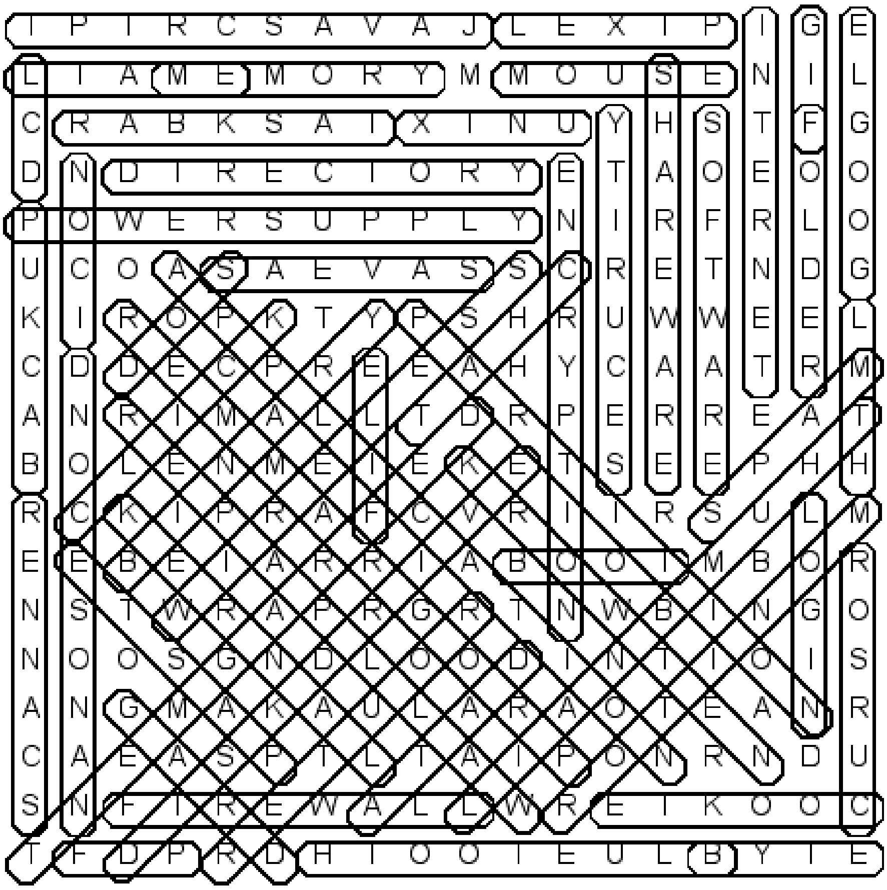
“Puzzling it out”

Due: Jan. 31, 2020 @ 4:00 pm (late date Feb. 3 @ 4:00 pm)

The objective of this assignment is to apply arrays in the development of a problem solution.

WordSearch Puzzle

A wordsearch puzzle is a square or rectangular grid of letters. Within the grid are a number of words that may be written right, left, up, down and diagonally up-right, up-left, down-right and down-left. The goal is to find all the words within the grid. Usually a list of words is supplied. A sample wordsearch puzzle (and its solution) is shown below:

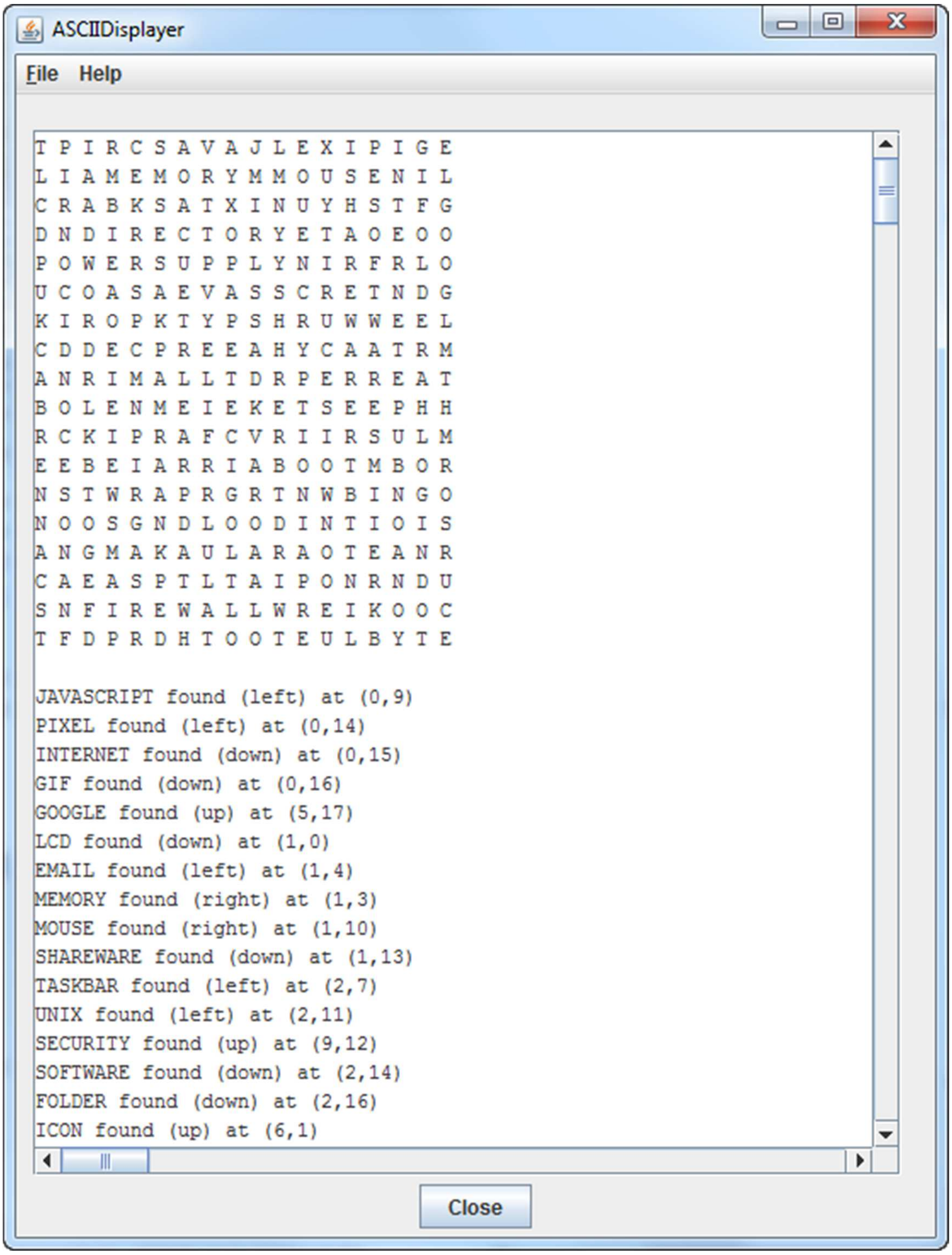


```
T P I R C S A V A J L E X I P I G E
L I A M E M O R Y M M O U S E N I L
C R A B K S A T X I N U Y H S T F G
D N D I R E C T O R Y E T A O E O O
P O W E R S U P P L Y N I R F R L O
U C O A S A E V A S S C R E T N D G
K I R O P K T Y P S H R U W W E E L
C D D E C P R E E A H Y C A A T R M
A N R I M A L L T D R P E R R E A T
B O L E N M E I E K E T S E E P H H
R C K I P R A F C V R I I R S U L M
E E B E I A R R I A B O O T M B O R
N S T W R A P R G R T N W B I N G O
N O O S G N D L O O D I N T I O I S
A N G M A K A U L A R A O T E A N R
C A E A S P T L T A I P O N R N D U
S N F I R E W A L L W R E I K O O C
T F D P R D H T O O T E U L B Y T E
```

For this assignment you will develop a solver for such puzzles.

The puzzle is prepared as an `ASCIIDataFile`. On the first line are two integers giving the height and width of the puzzle. Following that are *height* lines consisting of *width* characters (`char`), tab delimited, making up the puzzle. Finally, there are some number of lines, each containing a word (`String`), being the words that are to be located within the puzzle.

The output is written to an `ASCIIDisplayer` with text area: 35×70. The puzzle should first be displayed, followed by one line per word, indicating where the word as found in the puzzle. For example, the first part of the output for the above puzzle would look like:



```
T P I R C S A V A J L E X I P I G E
L I A M E M O R Y M M O U S E N I L
C R A B K S A T X I N U Y H S T F G
D N D I R E C T O R Y E T A O E O O
P O W E R S U P P L Y N I R F R L O
U C O A S A E V A S S C R E T N D G
K I R O P K T Y P S H R U W W E E L
C D D E C P R E E A H Y C A A T R M
A N R I M A L L T D R P E R R E A T
B O L E N M E I E K E T S E E P H H
R C K I P R A F C V R I I R S U L M
E E B E I A R R I A B O O T M B O R
N S T W R A P R G R T N W B I N G O
N O O S G N D L O O D I N T I O I S
A N G M A K A U L A R A O T E A N R
C A E A S P T L T A I P O N R N D U
S N F I R E W A L L W R E I K O O C
T F D P R D H T O O T E U L B Y T E

JAVASCRIPT found (left) at (0,9)
PIXEL found (left) at (0,14)
INTERNET found (down) at (0,15)
GIF found (down) at (0,16)
GOOGLE found (up) at (5,17)
LCD found (down) at (1,0)
EMAIL found (left) at (1,4)
MEMORY found (right) at (1,3)
MOUSE found (right) at (1,10)
SHAREWARE found (down) at (1,13)
TASKBAR found (left) at (2,7)
UNIX found (left) at (2,11)
SECURITY found (up) at (9,12)
SOFTWARE found (down) at (2,14)
FOLDER found (down) at (2,16)
ICON found (up) at (6,1)
```

Hints

- The puzzle can be represented as a right-sized two-dimensional array of characters (`char`).
- A `String` can be converted into a right-sized array of characters via the `String` method `toCharArray`.

- A word can occur in any of 8 directions starting from a square in the puzzle:
    - to the right (normal text: `MEMORY` above)
    - to the left (backwards: `PIXEL`)
    - downwards (`INTERNET`)
    - upwards (`GOOGLE`)
    - diagonally downwards to the right (`APPLICATION`)
    - diagonally downwards to the left (`CHAT`)
    - diagonally upwards to the right (`WIRELESS`)
    - diagonally upwards to the left (`PROGRAMMER`)
- Consider writing separate methods such as:

```
private boolean searchRight ( int x, int y, String word ) {
```

  that look for a `word` starting from position `(x,y)` in the puzzle and proceeding in the designated direction (right in this case). The method returns `true` if the word is found in the designated direction starting from the indicated position. You would have 8 such methods. Each of these methods will be quite similar.

- A word cannot occur starting from `(x,y)` if there isn't enough room from `(x,y)` to the edge of the puzzle in the designated direction. For example, if the puzzle is 10×10, we are checking from position `(3,7)` to the right and the word is more than 3 characters in length.
- Consider developing the solution in phases such as:
    1. read and display puzzle
    2. check for a single word in one direction only (e.g. right)
    3. repeatedly add the ability to check for a single word in each of the other 7 directions
    4. add the ability to search for multiple words.
- Don't use the big puzzle for testing. Make special small puzzles to test each of the 8 search methods above such as the included puzzle `right.txt` that has only one word written to the right.

Submission:

Details regarding preparation and submission of assignments in COSC 1P03 are found on the COSC 1P03 Sakai Site as `Assignment Guidelines` under `Course Documents`. This document includes a discussion of assignment preparation, programming standards, evaluation criteria and academic conduct (including styles for citation) in addition to the detailed assignment submission process copied below.

To prepare and submit the assignment electronically, follow the procedure below:

1. Ensure your folder (say `Assign_1`) for the assignment is accessible on your computer and contains the Dr. Java project and all associated `java` and `class` files for your assignment.

2. Create a `.zip` file of your submission by right-clicking on the top level folder (i.e. `Assign_1`) and selecting `Send to/Compressed (zipped) folder`. A zipped version of the folder will be created. Use the default name (`Assign_1.zip`). It is **important** that you only submit a `.zip` file, not `.rar` or `.tar` or any other type of compression. If you use a type of compression other than `.zip` your assignment may not be marked.

3. Log on to Sakai and select the COSC 1P03 site.
4. On the `Assignments` page select `Assignment 1`. Attach your `.zip` file (e.g. `Assign_1.zip`) to the assignment submission (use the `Add Attachments` button and select `Browse`). Navigate to where you stored your assignment and select the `.zip` file (e.g. `Assign_1.zip`). The file will be added to your submission. Be sure to check the `Honor Pledge` checkbox. Press `Submit` to submit the assignment.
5. Assignments incorrectly submitted will lose marks. Assignments without the required files may not be marked.

DrJava

The `.zip` folder you submit should contain the project folder including all files relevant to the project - the `.drjava`, `.java` and `.class` files for the assignment. If your project requires any special instructions to run, these instructions must be included in a read me file.

Other Platforms

Students must create their project using an IDE that is available on the Brock Computer Science lab computers. Currently, these are NetBeans, IntelliJ and Dr. Java. Markers must be able to open, compile and run your project on the lab computers. Assignments completed using some other IDE may not be marked.