# Project 1 Part C: GuessPhrasePanel & RandomString

The goal of this project is to become more familiar with creating and using classes and objects in Java, to further introduce you to the concepts of abstraction and modularization, and to directly use collections of items. You should begin with the Hangman project you’ve been working on.

# Instructions

1. Create a RandomString class and implement the following:
	1. Create a file named guess\_phrases.txt in your project that contains phrases to be guessed in your Hangman game. This file will have one guess phrase per line. Make sure this file is in the topmost project folder, ***not*** inside your src folder.
	2. A constructor that receives the name of a file to get string values from. The constructor should read in the phrases from the file and store them for later use.
	3. next – a method that returns a random string value from the file; this value shouldn’t be repeated until all guess phrases in the file have been used.
	4. Create a main method to test that next is working correctly by repeatedly calling next & printing the result – you should not have any repeats, and the phrases should *not* be in the same order as in the file.
2. Create a GuessPhrasePanel class and implement the following (NOTE: Make sure your methods work the same for upper and lower case letters):
	1. Has a constructor taking a String guess phrase
	2. Shows all the letters in the phrase, initially just showing the underline for all alphabet letters (not the letter as it needs to be guessed). Spaces and other punctuation should be shown with no underline.
	3. setPhrase – sets a new phrase to this GuessPhrase object, gets rid of old phrase parts and creates new ones
	4. hasLetter – a method that returns whether or not a specified letter is in the phrase
	5. revealLetter – a method that reveals or shows a specified letter (if there are more than one of that letter, all should be revealed)
	6. isFullPhraseRevealed – returns whether or not the full phrase has been revealed
	7. revealFullPhrase – reveals the full phrase
3. Create a HangmanGame class to test GuessPhrasePanel
	1. Create fields for RandomString and GuessPhrasePanel objects
	2. Create a constructor that initializes the fields and adds a GuessPhrase panel to a JFrame. The constructor should also add a KeyListener to the frame that does the following:
		1. reveals the letter typed if it is in the guess phrase
		2. reveals the whole phrase if the enter key is pressed
		3. gets a random string and sets it as the text for the GuessPhrase if the space bar is pressed
		4. does nothing if any other key is pressed
	3. Create a main that calls the HangmanGame constructor

# Hints / tips

* When initializing your GuessPhrasePanel and creating the setPhrase method, you may find it useful to create a private initializeLetters() method to be used by both the constructor and setPhrase. It would look like:
**private** **void** initializeLetters() {

 **this**.removeAll(); // throw away existing letters in phrase

 **this**.updateUI(); // reset entire UI

 // add code statements to

// initialize letters in phrase to be guessed
 // & add them to the GuessPhrasePanel

**this**.revalidate();// push changes through
}

* You can match special keys on the keyboard like enter or space using virtual key constants in the KeyEvent class, like KeyEvent.VK\_ESCAPE.