Project 1

Part I Join the GitHub Organization

In this project, you will be working with other students as a team. A team can have a maximum of 3 members, and one member has to take the role of the team leader. A team leader will:

- initiate meetings
- make sure every member takes a fair share of tasks
- lead the team to beat the deadline.

The first thing you need to do is to confirm the invitation to join our GitHub organization. Depending on the lecture time you registered for, you should join https://github.com/orgs/wwu-csci-145.

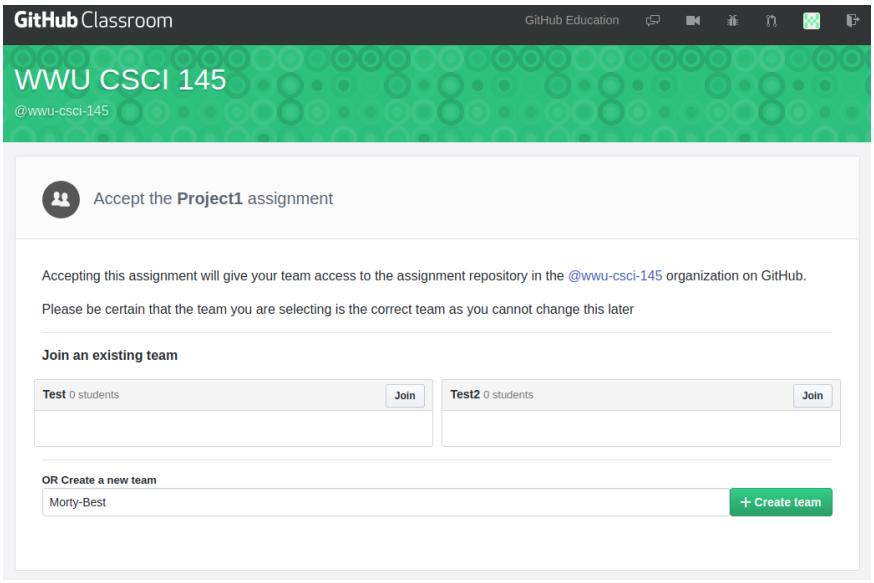
You should receive an invitation to join the corresponding GitHub organization yesterday. If not, you should log into GitHub, click the above corresponding link, and click the invitation shown up on the top of the screen.

However, this may not work for you for many reasons. For instance, we noticed that some GitHub usernames provided to us were invalid. In this case, please fill the one-question survey titled "**GitHub Username (if you did not receive an invitation from us)**" on Canvas. This may stop you from continuing working on this assignment, so we will add you to the GitHub organization on a rolling basis, and we expect you to act fast as well.

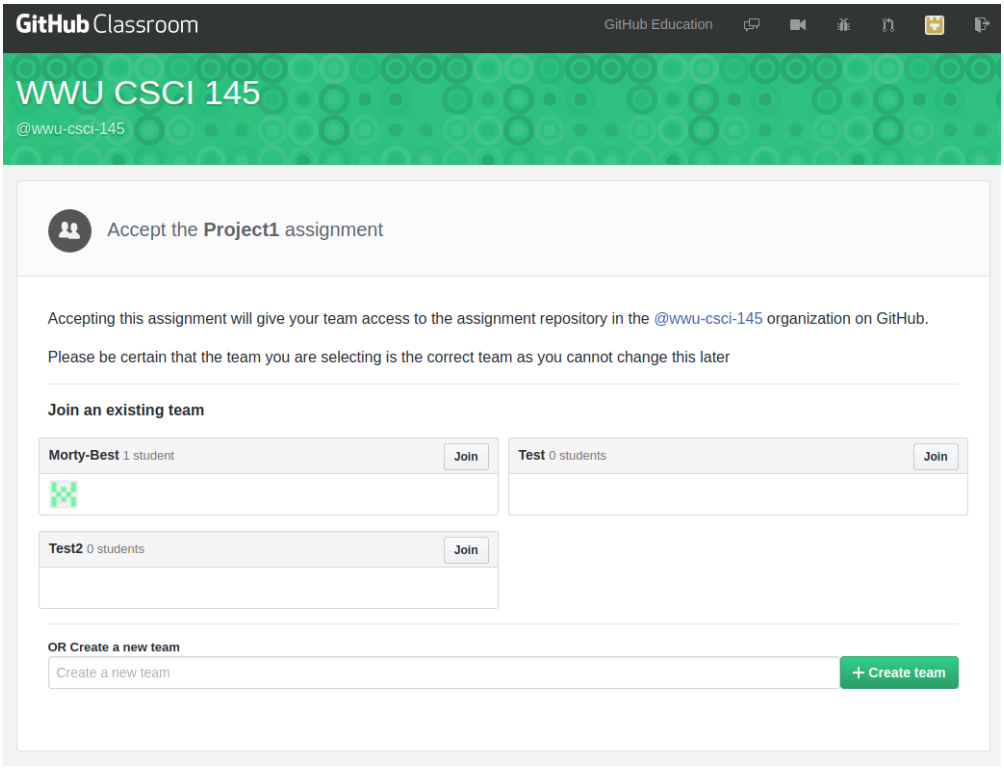
Part II Set up a Team

You should have a team first. If you did not get a chance to know others in class, you can post on Piazza to ask for others to join your team, but you do want to make sure to list your lecture section clear in your post. **People who registered different lecture time should NOT be on the same team**, and our GitHub organization configuration will stop you from doing so.

If you are a team leader, you will need to initialize a repository for your team first. When you click the provided link, you will be asked to give a name to your team. In the following example, you will see that the team is named "Morty-Best".



If you are a team member, you will need to confirm that your team leader has done the above step. After that, you will need to find your team among possibly many listed teams, and join it. In the following example, we will join the team named "Morty-Best".



Note that the above steps are based on the assumption that you have joined one of the GitHub organizations. If you never did this before, see Section I.

Depending on which lecture you registered for, you need to click different links. Note that if you are a team member, you should wait till your team leader to initiate a team firstly:
https://classroom.github.com/g/5Xvg_xwP

Part III Hangman

In this project, you are going to implement the word guessing game, Hangman. Your program is going to get a random word from an enumerated list (instructions below) and then let the user guess letters until they get the word correct, run out of guesses, or solve the word. A unique characteristic of Hangman is that the user must specify a subset of the available spaces to check. So, for each guess, the user provides the letter to guess along with the spaces he/she wants to check.

The game has three levels of difficulty. The difficulty level determines the number of spaces allowed to check on each guess along with the total number of guesses the user can make. At the easy difficulty level, the user is allowed 15 guesses and must specify 4 spaces to check per guess. Users playing at the intermediate difficulty level get 12 guesses and must specify 3 spaces. When the game is on the hard difficulty level, the user gets 10 guesses and must specify 2 spaces at a time. An incorrect guess (or solve attempt) causes the program to decrement the number of remaining guesses, but a correct guess does not affect the number of remaining guesses. Also, invalid input does not affect the number of remaining guesses.

Continue to finish the Java program named **Hangman.java** that allows the user to play Hangman as many times as they wish as long as the user doesn't exceed 20 games (the maximum number of games the user can play – that's how many items are in the word bank). When the program begins, it will ask for the difficulty level from the user. The program should read the user's input as a String, take the first letter from the string, and make sure that it is equal to 'e', 'i', or 'h'. If the input is invalid, then a message is printed, and the user is re-prompted until the input is valid. The program then asks the user for their guess. At any point, the user can choose to "solve" the word instead of entering another level. If the user enters "solve" with any capitalization, a message should be printed allowing the user to solve the game. If this answer is incorrect, the number of guesses should be decremented and the user should be re-prompted to enter a letter. If the user correctly solves the word, the game should end. After the game ends, the program will print the result and ask the user if they want to play again. Many other scenarios are given in the

examples below. Your program should behave the same for all test cases given. **You should also come up with additional test cases in order to thoroughly test your application**.

Each example has the output of a single run of a correctly working program. In these examples, testingMode is on, which means that you can see the secret word before guessing (see below for more details). This is very handy when debugging your code. The user's input in the examples is marked in green.

Example 1 (User wins by guessing letters):

```
Enter your difficulty: Easy (e), Intermediate (i), or Hard (h)
eeeeeeeeee
The secret word is: identifier
The word is: ----------
Please enter the letter you want to guess: d
Please enter the spaces you want to check (separated by spaces):
0 1 2 5
Your guess is in the word!
The updated word is: -d--------
Guesses Remaining: 15
Please enter the letter you want to guess: i
Please enter the spaces you want to check (separated by spaces):
0 1 5 7
Your guess is in the word!
The updated word is: id---i-i--
Guesses Remaining: 15
Please enter the letter you want to guess: e
Please enter the spaces you want to check (separated by spaces):
2 3 8 9
Your guess is in the word!
The updated word is: ide--i-ie-
Guesses Remaining: 15
Please enter the letter you want to guess: n
Please enter the spaces you want to check (separated by spaces):
3 4 5 6
Your guess is in the word!
The updated word is: iden-i-ie-
Guesses Remaining: 15
Please enter the letter you want to guess: t
Please enter the spaces you want to check (separated by spaces):
3 4 5 6
Your guess is in the word!
The updated word is: identi-ie-
Guesses Remaining: 15
Please enter the letter you want to guess: f
```

```
Please enter the spaces you want to check (separated by spaces):
4 5 6 7
Your guess is in the word!
The updated word is: identifie-
Guesses Remaining: 15
Please enter the letter you want to guess: r
Please enter the spaces you want to check (separated by spaces):
6 7 8 9
Your guess is in the word!
The updated word is: identifier
Guesses Remaining: 15
You have guessed the word! Congratulations
Would you like to play again? Yes(y) or No(n)
y
Enter your difficulty: Easy (e), Intermediate (i), or Hard (h)
…… // execution continues for the next round
```

Example 2 (Checking invalid input)

```
Enter your difficulty: Easy (e), Intermediate (i), or Hard (h)
f
Invalid difficulty.  Try Again...
Enter your difficulty: Easy (e), Intermediate (i), or Hard (h)
g
Invalid difficulty.  Try Again...
Enter your difficulty: Easy (e), Intermediate (i), or Hard (h)
e
The secret word is: identifier
The word is: --------
Please enter the letter you want to guess: 7
Your input is not valid.  Try again.
Guesses Remaining: 15
Please enter the letter you want to guess: a
Please enter the spaces you want to check (separated by spaces):
1 2
Your input is not valid.  Try again.
Guesses Remaining: 15
Please enter the letter you want to guess: a
Please enter the spaces you want to check (separated by spaces):
1 2 3
Your input is not valid.  Try again.
Guesses Remaining: 15
Please enter the letter you want to guess: a
Please enter the spaces you want to check (separated by spaces):
1 2 3 4 5
Your input is not valid.  Try again.
```

```
Guesses Remaining: 15
Please enter the letter you want to guess: b
Please enter the spaces you want to check (separated by spaces):
1 5 9 12
Your input is not valid.  Try again.
Guesses Remaining: 15
…… //Execution continues from this point.
```

Example 3 (If two letters are entered as the guess, take the first one)

```
Enter your difficulty: Easy (e), Intermediate (i), or Hard (h)
e
The secret word is: identifier
The word is: ----------
Please enter the letter you want to guess: id
Please enter the spaces you want to check (separated by spaces):
0 1 2 3
Your guess is in the word!
The updated word is: i---------
Guesses Remaining: 15
Please enter the letter you want to guess://Execution continues beyond this
point.
```

Example 4 (If user makes an identical correct guess or a guess containing an already uncovered space, don't decrement guesses remaining)

```
Enter your difficulty: Easy (e), Intermediate (i), or Hard (h)
e
The secret word is: identifier
The word is: ----------
Please enter the letter you want to guess: i
Please enter the spaces you want to check (separated by spaces):
0 0 0 0
Your guess is in the word!
The updated word is: i---------
Guesses Remaining: 15
Please enter the letter you want to guess: i
Please enter the spaces you want to check (separated by spaces):
0 1 2 3
Your guess is in the word!
The updated word is: i---------
Guesses Remaining: 15
Please enter the letter you want to guess: i
Please enter the spaces you want to check (separated by spaces):
1 2 3 4
Your letter was not found in the spaces you provided.
Guesses Remaining: 14
```

```
Please enter the letter you want to guess: i
Please enter the spaces you want to check (separated by spaces):
3 4 5 6
Your guess is in the word!
The updated word is: i----i----
Guesses Remaining: 14
Please enter the letter you want to guess: //Execution continues from this
point
```

Example 5 (Max guesses is exceeded, resulting in a player loss)

```
Enter your difficulty: Easy (e), Intermediate (i), or Hard (h)
h
The secret word is: identifier
The word is: ----------
Please enter the letter you want to guess: z
Please enter the spaces you want to check (separated by spaces):
0 1
Your letter was not found in the spaces you provided.
Guesses Remaining: 9
Please enter the letter you want to guess: z
Please enter the spaces you want to check (separated by spaces):
1 2
Your letter was not found in the spaces you provided.
Guesses Remaining: 8
Please enter the letter you want to guess: z
Please enter the spaces you want to check (separated by spaces):
4 5
Your letter was not found in the spaces you provided.
Guesses Remaining: 7
Please enter the letter you want to guess: q
Please enter the spaces you want to check (separated by spaces):
0 3
Your letter was not found in the spaces you provided.
Guesses Remaining: 6
Please enter the letter you want to guess: r
Please enter the spaces you want to check (separated by spaces):
0 1
Your letter was not found in the spaces you provided.
Guesses Remaining: 5
Please enter the letter you want to guess: z
Please enter the spaces you want to check (separated by spaces):
y
Your input is not valid. Try again.
Guesses Remaining: 5
Please enter the letter you want to guess: y
Please enter the spaces you want to check (separated by spaces):
```

```
2 3
Your letter was not found in the spaces you provided.
Guesses Remaining: 4
Please enter the letter you want to guess: h
Please enter the spaces you want to check (separated by spaces):
4 5
Your letter was not found in the spaces you provided.
Guesses Remaining: 3
Please enter the letter you want to guess: d
Please enter the spaces you want to check (separated by spaces):
7 8
Your letter was not found in the spaces you provided.
Guesses Remaining: 2
Please enter the letter you want to guess: u
Please enter the spaces you want to check (separated by spaces):
0 3
Your letter was not found in the spaces you provided.
Guesses Remaining: 1
Please enter the letter you want to guess: a
Please enter the spaces you want to check (separated by spaces):
3 4
Your letter was not found in the spaces you provided.
Guesses Remaining: 0
You have failed to guess the word... :(
Would you like to play again? Yes(y) or No(n)
n
```

Example 6 (Solving word)

```
Enter your difficulty: Easy (e), Intermediate (i), or Hard (h)
i
The secret word is: identifier
The word is: ----------
Please enter the letter you want to guess: i
Please enter the spaces you want to check (separated by spaces):
0 4 5
Your guess is in the word!
The updated word is: i----i----
Guesses Remaining: 12
Please enter the letter you want to guess: solve
Please solve the word: identical
That is not the secret word.
Guesses Remaining: 11
Please enter the letter you want to guess: d
Please enter the spaces you want to check (separated by spaces):
```

```
0 1 2
Your guess is in the word!
The updated word is: id---i----
Guesses Remaining: 11
Please enter the letter you want to guess: solve
Please solve the word: identifier
You win!
You have guessed the word! Congratulations
Would you like to play again? Yes(y) or No(n)
n
```

Design Requirements

After working on lab3, you might be familiar with the concept of decoupling. The same concept will be applied to your project 1 as well. You will be required to create three methods:

- isInt: check if a string can be parsed to an int
- validPosition: check if provided spaces are valid
- getPosition: convert valid spaces from a string to an int array

The definition and examples of the methods are provided in the template of your project 1. **You should NOT change the method names, return data types and parameters per method. However, you are welcome to create more methods if needed.**

Other Requirements

You **must** use index 0 as the index of the first character in your word. For example, if the secret word is "hardware", the letter 'h' is at index 0. As such, if the user specifies space 0, you should uncover the 'h'. Failure to use correct indexing will have a negative impact on your assignment grade.

In order to facilitate the testing of your program, you **must** include a boolean variable called testingMode initialized to true at the top of your class (under the class declaration and above your main method) as shown below.

```
private static final boolean testingMode = true;
```

If the value of the variable testingMode is true, your program will display the secret word the user should guess. This helps you test your program and helps us grade your program. Without this variable, we will not know what the correct answer is. See

example 1 to see the output. On the other hand, if the value of the variable testingMode is set to false, the program will not show the value of the secret word. **When you submit your code, make sure the variable is set to true.**

Use the RandomWord.newWord() method to generate random words, and this method should only be called once per game, and it can be called at most 20 times in a single program run (note: after 20 calls, it issues an error message and terminates). This method is provided for you. However, you must download the RandomWord.java file from the labs and projects website, and place RandomWord.java in the same source directory as Hangman.java. RandomWord.java randomly picks a word from an enumerated list and returns it when the newWord() method is called. **Failure to use this method may result in a failing grade**.

Hints

You may want to use two Strings to store the secret word and the word that is displayed to the user. The secret word is stored internally (the user can't see it unless testingMode is set to true) and the display word is what they see on the screen. The displayed word will start out as a series of dashes ("-") and then be modified after a correct guess.

You may find the following methods useful:

- Character.isLetter(*char arg*) – returns true if the character arg is a letter
- Character.isDigit(*char arg*) – returns true if the character arg is a number
- Character.getNumericValue(*char arg*) – returns the numeric value of a character

Project Submission and Grading

Please use comments to explain your program or any sections that are possibly difficult to understand. 10% of the project points would be deducted if few or no comments can be found in your java files. If your java files fail to compile, you may risk getting a 0.

You should check https://travis-ci.com, and find your project repository. If your code fails to pass the testing cases on Travis-CI, it is not likely you will get a good grade. **However, even if you pass all test cases on Travis-CI, you may still not earn a good grade if you do not test your code thoroughly.**