FULL SAIL UNIVERSITY

COP2334

PG2 – LAB: HISTOGRAM

CONTENTS

FULL SAIL UNIVERSITY

COP2334

PART A – READ METHODS, MENU LOOP

Part A-1: Console Application

SETUP

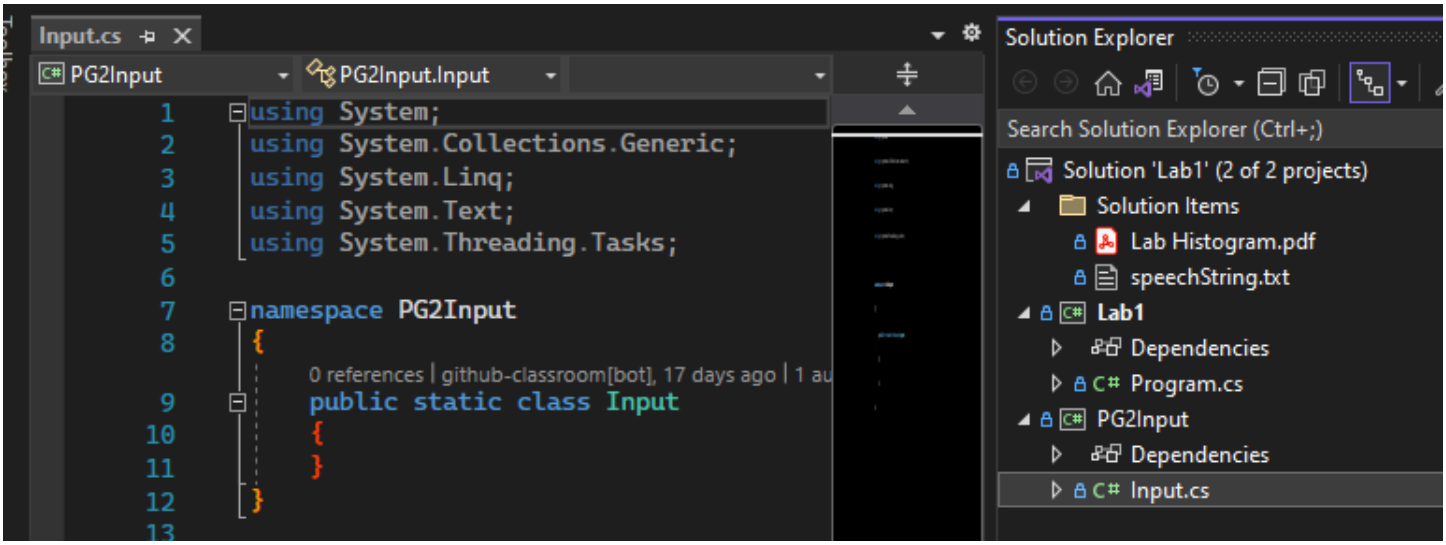
A **C# .NET Core console application** has been provided for you in your **GitHub repo**. **Use the provided solution.**

Input Class

Lab Overview Video

Overview

A static class called Input has been provided in the PG2Input project. **Put the Part A methods inside of the Input class.**



GRADING: 5 POINTS

Part A-2: GetInput

Lecture Videos

Method Basics Examples 02
Method Basics Challenge 02

Create a method called **GetInput** that will ask the user to input something. The method should show the message parameter, read the user's input, and return the input**.**

NAME	RETURNS	PARAMETERS	COMMENTS
GetInput	string	message	Show the message, get input from the user, return the input

EXAMPLE USAGE

These are examples of how you could call the method once you've written the code for it. NOTE: they are just examples. It is not to be used exactly as is for the lab.

```
String input = Input.GetInput("What is your name?);
```

GRADING: 2 POINTS

Part A-3: ValidInteger

Create a method called **ValidInteger** that will return true/false if the integer that is passed in is within the min and max range (inclusively)**. NOTE: the number, min and max are parameters that are passed in to the method.**

NAME	RETURNS	PARAMETERS	COMMENTS
ValidInteger	bool	number min max	Return if the number parameter is within the min and max range

EXAMPLE USAGE

These are examples of how you could call the method once you've written the code for it. NOTE: they are just examples. It is not to be used exactly as is for the lab.

```
int number = 5;
bool isValid = Input.ValidInteger(number, 10, 20); //ValidInteger should return false
isValid = Input.ValidInteger(number, 1, 100); //ValidInteger should return true
```

GRADING: 3 POINTS

FULL SAIL UNIVERSITY

COP2334

Part A-4: GetInteger

Lecture Videos

Method Basics Examples 02
Method Basics Challenge 02
Converting Strings
Converting Strings Example
Converting Strings Challenge

Lab Overview Video

Overview

Create a method called **GetInteger** that will return an integer that fits within a min/max range**.** Call **GetInput** to get the user's input. Convert the string that is returned from GetInput to an integer. **DO NOT THROW AN UNHANDLED EXCEPTION.** If the input is a number, then call **ValidInteger** to check if the number is within the min/max range. Return the number if it is valid. If the user's input is not a number OR the number is not valid, print an error message. **Do not return until the user enters a valid integer**. Therefore, you'll need a **loop**.

NAME	RETURNS	PARAMETERS	COMMENTS
GetInteger	int	message min max	Return a valid integer

EXAMPLE USAGE

These are examples of how you could call the method once you've written the code for it. NOTE: they are just examples. It is not to be used exactly as is for the lab.

```
int year = Input.GetInteger("Year: ", 1908, 2021);
```

EXAMPLE OUTPUT

Year: **steve**
That is not an integer. Please try again.
Year: **2019**

GRADING: 5 POINTS

COMMON MISTAKES:

- -2: Calling int.Parse after already calling int.TryParse. If you call int.TryParse and it returns true, then the string is converted and the number is stored in the out parameter.
- -2: using int.Parse without a try-catch. GetInteger should not throw an unhandled exception. Catch the exception using a try-catch and show a message to the user. Continue looping until the input is valid.
- -2: Calling the GetInteger method recursively. A simple loop is better in this scenario so do not use recursion.
- -2: Not calling GetInput or ValidInteger

Part A-5: ValidString

Create a method called **ValidString** that will return true/false if the string that is passed in is not null and not empty**.** You should use the IsNullOrEmpty or the IsNullOrWhiteSpace methods of the string class to check if the string is empty.

NAME	RETURNS	PARAMETERS	COMMENTS
ValidString	bool	value	Return if the value parameter is not null and not empty

EXAMPLE USAGE

These are examples of how you could call the method once you've written the code for it. NOTE: they are just examples. It is not to be used exactly as is for the lab.

```
string input = "Steve";
bool isValid = Input.ValidString(input); //ValidInteger should return true
isValid = Input.ValidString(""); //ValidInteger should return false
```

GRADING: 3 POINTS

Part A-6: GetString

Lecture Videos

Parameters By Reference
Parameters By Reference Example
Parameters By Reference Challenge

Lab Overview Video

Overview

Create a method called **GetString** that will ask the user for a string. Instead of returning the value like in GetInteger, you should use **pass by reference** to get the string back to the caller. Call **GetInput** to get the user's input and store the input in the ref parameter. Call **ValidString** to check if the input is valid. Return if it is valid. If the user's input is not valid, print an error message. **Do not return until the user enters something.** Therefore, you'll need a **loop**.

NAME	RETURNS	PARAMETERS	COMMENTS
GetString	void	message value	Return a valid string through the ref parameter (value).

EXAMPLE USAGE

These are examples of how you could call the method once you've written the code for it.

```
string make = string.Empty;
Input.GetString("What is the make of your car: ", ref make);
```

EXAMPLE OUTPUT

What is the make of your car: **Ford**

GRADING: 7 POINTS

COMMON MISTAKES:

- -1: converting the string input to a number as part of validation. The only validation you need to check is whether the input is empty or not.
- -2: returning without checking if the input is empty. GetString should not return if the user's input is empty.
- -2: Not calling GetInput or ValidString

Part A-7: GetMenuChoice

Lecture Videos

Out Parameters
Out Parameters Example
Out Parameters Challenge

Lab Overview Video

Overview

Create a method called **GetMenuChoice** that will ask the user to select from a list of options. Instead of returning like GetInteger or passing back the value like GetString, you should return the menu selection through an **out parameter**. The method should show a list of options to the user (the menuOptions parameter). Get the user's selection by calling GetInteger and assign the integer to the **out parameter**.

You'll need to pass the list of options as a **string array**. Something like string[] {"1. Add Car", "2. Show Cars", "3. Exit" }. The method should loop over the array and show each option on a new line.

NAME	RETURNS	PARAMETERS	COMMENTS
GetMenuChoice	void	message menuOptions menuSelection	Return the menu selection through the out parameter (menuSelection).

EXAMPLE USAGE

These are examples of how you could call the method once you've written the code for it.

```
int menuChoice = 0;
string[] mainMenu = new string[] { "1. Add Car", "2. Show Cars", "3. Exit" };
Input.GetMenuChoice("Choice? ", mainMenu, out menuChoice);
```

EXAMPLE OUTPUT

1. Add Car
2. Show Cars
3. Exit

Choice? **Steve**
That is not a number. Please try again.
Choice? **2**

GRADING: 5 POINTS

COMMON MISTAKES:

- -3: duplicating the GetInteger logic. GetMenuChoice should call GetInteger instead of duplicating the code.
- -1: hardcoding the max passed to GetInteger. You should use the options.Length for the max value passed to GetInteger.
- -1: incorrect min passed to GetInteger.
- -3: creating the array of options inside the method. GetMenuChoice should just print the items in the array parameter.

Part A-8: Menu loop

Lecture Videos

Input Challenge
Converting Strings Challenge

Lab Overview Video

Overview

You will need to create a loop in **Main** that handles the menu options for lab 1. This should be a simple **while** loop that loops while the menu selection is NOT exit. ***Inside*** the while loop, you should 1) call **GetMenuChoice** to show the menu and get the user's menu selection. 2) use a **switch** statement that has logic for each menu option.

NOTE: for Part A, you will only need code to handle the exit option in the switch.

The Menu to show:

1: The Speech
2: List of Words
3: Show Histogram
4: Search for Word
5: Remove Word
6: Exit

GRADING: 5 POINTS

COMMON MISTAKES:

- -2: Exit does not exit.

FULL SAIL UNIVERSITY

COP2334

PART B – THE LIST

Part B-1: The Speech

Lecture Videos

Method Basics Examples 02
Method Basics Challenge 02

Lab Overview Video

Overview

NOTE: the data to use for this project in the **speechString.txt** file for the lab. The file is in the Lab1 folder in your repo.

Create a method called **GetSpeech**. **Copy and paste the text from the file to the method.** The method should simply return the string.

Call the GetSpeech method from Main. Do this once and BEFORE the menu loop and store the result in a string variable.

NAME	RETURNS	PARAMETERS	COMMENTS
GetSpeech	String	(none)	Returns the string that is supplied in the speechString.txt file.

Add code to the first menu option to show the speech.

GRADING: 5 POINTS

COMMON MISTAKES:

- -3: not creating a GetSpeech method
- -2: trying to read the file instead of copy and pasting the text into the method.
- -2: not showing the speech
- -2: calling the GetSpeech multiple times
- -1: not storing the returned string in a variable in Main

FULL SAIL UNIVERSITY

COP2334

Part B-2: Splitter

Lecture Videos

Splitting Strings Channel
List Basics
List Basics Example
List Basics Challenge

Lab Overview Video

Overview

Create a method called **Splitter** that will **split the string parameter** into an **array of words** using the **delimiters** parameter. Make sure you remove the empty entries when splitting. Convert the array of words to a list of strings and return the list.

Before the menu loop in Main, call Splitter to get your list of words from the speech. Pass the string returned from GetSpeech to the Splitter method.

Add code to the second menu option to show the list of words. Clear the screen then print each word in the list on a separate line.

NAME	RETURNS	PARAMETERS	COMMENTS
Splitter	List<string>	text delimiters	Using the character array of delimiters, split the text parameter, convert the string array to a list, then return the list. Make sure you remove the empty entries when splitting.

EXAMPLE USAGE

These are examples of how you could call the method once you've written the code for it.

```
char[] delimiters = new char[] { '^', '$' };
string stringToSplit = "Batman^The Bat^Robin&The Boy Wonder";
List<string> heroes = Splitter(stringToSplit, delimiters);
```

GRADING: 5 POINTS

COMMON MISTAKES:

- -3: not creating a Splitter method
- -2: not splitting on the correct delimiters. To get the words, you need all punctuation and the escape sequences (\n, \t, \r) in your list of delimiters.
- -3: not converting the string array to a List. The Split method returns an array of strings. Convert that to a List<string>.
- -2: not using StringSplitOptions.RemoveEmptyEntries
- -1: not storing the returned list in a variable in Main

FULL SAIL UNIVERSITY

COP2334

Part B-3: List of Words

Lecture Videos

List Looping
List Looping Example
List Looping Challenge

Lab Overview Video

Overview

Before the menu loop in Main, call Splitter to get your list of words from the speech. Pass the string returned from GetSpeech to the Splitter method. Store the returned list in a variable.

Add code to the second menu option to show the list of words. Clear the screen then print each word in the list on a separate line.

GRADING: 5 POINTS

COMMON MISTAKES:

- -2: not using the string -2: not calling Splitter before the menu
- -2: not printing each word in the list
- -2: not using the string from part B-1

FULL SAIL UNIVERSITY

COP2334

PART C – THE DICTIONARY

Part C-1: Word counts

Lecture Videos

List Looping
List Looping Example
List Looping Challenge
Dictionary Creating Adding
Dictionary Examples
Dictionary Basics Challenge
Dictionary Checking Keys
Dictionary Checking Keys Example
Dictionary Updating Values
Dictionary Updating Values Example
Dictionary Keys Values Challenge

Lab Overview Video

Overview

Create a method called SpeechCounts. Now that you have the **list of words**, you need to calculate how many times each word appears in the list of words. In the method, create a **Dictionary** to store those counts. The key of the dictionary will be the words and the value will be the counts. Loop over the **List of words** and ***put*** or ***update*** the word in the dictionary.

Call SpeechCounts BEFORE the menu loop. Store the dictionary in a variable to use for the menu options.

NOTES:

- Make it **case-insensitive** meaning that if the word is upper-case and lower-case in the data, only 1 will appear in the dictionary. For example, 'The' and 'the' are the same word so only one should be in the dictionary. **HINT: look at the different constructors for Dictionary to make this easier**.
- You need to check if the word is in the dictionary to decide if it should be **added** or if it should be **updated**. Use **ContainsKey** or **TryGetValue**.

NAME	RETURNS	PARAMETERS	COMMENTS
SpeechCounts	Dictionary	List of words	Using the list of words pulled from the speech, create a dictionary and calculate the counts for each word. Return the dictionary.

GRADING: 15 POINTS

COMMON MISTAKES:

- -3: not creating a SpeechCounts method
- -5: not calculating the word counts correctly. Loop over the list of words from the speech. Check if the word is in the dictionary and update the value if it is or add it if it is not.
- -5: using a list to store unique words. The keys in the dictionary are unique so no other list is needed outside of the list of words for the speech.
- -3: not making the keys case-insensitive (either through the dictionary or using ToUpper/ToLower).
- -2: not calling SpeechCounts before the menu loop
- -1: not storing the dictionary returned from SpeechCounts in a variable

Part C-2: PrintKeyValueBar

Lecture Videos

Output
Output Example
Output Challenge 1
Output Challenge 2
Output Challenge 3
Output Challenge 4
Output Challenge 5

Lab Overview Video

Overview

Create a method called **PrintKeyValueBar** that will print a word, a bar, and a count. The size of the bar should equal the count.

Note: the bar should be printed at a fixed horizontal position in the console so that the bars align properly in the chart. Use Console.CursorLeft to align the bars.

NAME	RETURNS	PARAMETERS	COMMENTS
PrintKeyValueBar	Void	word count	Prints the word, a bar, and the count

EXAMPLE OUTPUT:

GRADING: 5 POINTS

COMMON MISTAKES:

- -3: not creating a PrintKeyValueBar method
- -3: not showing the bar
- -2: not drawing the bar at a fixed horizontal position

Part C-3: Show Histogram

Lecture Videos

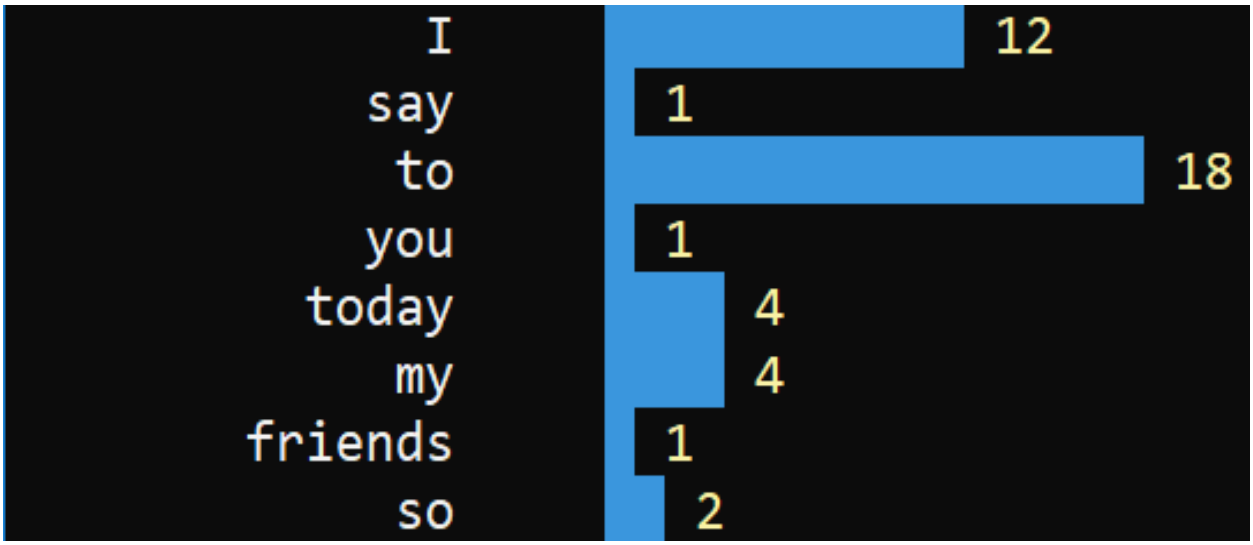
Dictionary Looping
Dictionary Looping Example
Dictionary Looping Challenge

Lab Overview Video

Overview

Add code to the **"Show Histogram"** menu option. For each word in the dictionary, call the **PrintKeyValueBar** method. **Pass the key and value as parameters to the method**.

EXAMPLE OUTPUT:



GRADING: 5 POINTS

COMMON MISTAKES:

- -2: duplicating the bar drawing logic instead of calling PrintKeyValueBar

Part C-4: Search for Word

Lecture Videos

Dictionary Checking Keys
Dictionary Checking Keys Example

Lab Overview Video

Overview

Add code for the **"Search for Word"** menu option. **Ask the user for a word to search for** ("What word do you want to find? "). **Use GetString to get the word from the user!**

If the word is in the Dictionary, print the word, bar, and count. **NOTE: call the PrintKeyValueBar method.**

GRADING: 10 POINTS

COMMON MISTAKES:

- -2: using a loop to find the search word in the dictionary. Using a loop to find a key in a dictionary defeats the purpose of using a dictionary. Use one of the build-in methods (ContainsKey or TryGetValue) instead.
- -2: trying to access a key's value without first checking if the key exists.
- -2: the bar is missing from the output.
- -1: not using GetString to get the search word from the user
- -2: not calling the PrintKeyValueBar
- -2: duplicating the bar drawing logic when searching for a word

Part C-5: Sentences for Word

Lab Overview Video

Overview

Add code to the "Search for Word" menu option.

Show the sentences that the word appears in. HINT: you can split the original speech text on different delimiters to get the sentences. Call the Splitter method you created earlier to get the list of sentences.

If the word is NOT in the dictionary, print "<word> is not found.". (replace <word> with what the user entered)

Example output:

```
1. Show Histogram
2. Search for Word
3. Exit
Choice? 2

Search word to find? friends

        friends   1
I say to you today, my friends, so even though we face the difficult
```

GRADING: 10 POINTS

COMMON MISTAKES:

- -3: using .Contains/.IndexOf to find the search word in a sentence string. They will give you false positives (Example: trying to find "any" using Contains will match with "anywhere").
- -2: not showing all the sentences for the word

FULL SAIL UNIVERSITY

Part C-6: Remove Word

Lecture Videos

Dictionary Removing
Dictionary Removing Example
Dictionary Removing Challenge

Lab Overview Video

Overview

Ask the user for a word to remove. **Use GetString to get the word to remove**.

Remove the word from the dictionary. If the word is not in the dictionary, show "<word> is not found". (replace <word> with what the user entered). **Do not use ContainsKey or TryGetValue.**

GRADING: 5 POINTS

COMMON MISTAKES:

- -1: not using GetString to get the search word from the user
- -2: using ContainsKey or TryGetValue before calling Remove
- -2: looping over the dictionary to find the key to remove
- -2: you need to check the result of calling Remove to print the proper message

FULL SAIL UNIVERSITY

COP2334

RUBRIC

Part A

FEATURE	VALUE
Part A-1: Console Application	5
Part A-2: GetInput	2
Part A-3: VaidInteger	3
Part A-4: GetInteger	5
Part A-5: ValidString	3
Part A-6: GetString	7
Part A-7: GetMenuChoice	5
Part A-8: Menu Loop	5
TOTAL	35

Part B

FEATURE	VALUE
Part B-1: The Speech	5
Part B-2: Splitter	5
Part B-3: List of Words	5
TOTAL	15

Part C

FEATURE	VALUE
Part C-1: Word Counts	15
Part C-2: PrintKeyValueBar	5
Part C-3: Show Histogram	5
Part C-4: Search for word	10
Part C-5: Sentences	10
Part C-6: Remove Word	5
TOTAL	50

FULL SAIL UNIVERSITY

COP2334

PROGRAMMER'S CHALLENGE

As with every programmer's challenge, remember the following…

1. Do the rubric first. Make sure you have something to turn in for the assignment.
2. When attempting the challenge, don't break your other code.
3. You have other assignments so don't sacrifice them to work on the challenges.

List Challenge

It would be nice to see the histogram data displayed in a sorted way. Two ways that would be interesting:

- Sort the word data alphabetically by word
- Sort the word data by the count

When selecting "Show Histogram", ask the user which to sort on then show the sorted word data chart.

The challenge is to sort the list of words by either the word or the count. Sorting by the words in a list should be easy. How would you sort by count? Those are stored in the dictionary.