



Assignment 1: Collections

Due Date: 12 April 2019

Weighting: 30%

Semester 1, 2019

Instructions

This assignment is an individual assignment.

NOTE: It is your responsibility to clarify any aspect of the assignment that you are unsure of, with your lecturer.

You are required to upload a single zip file which must contain multiple project solutions to Moodle before due date. The assignment will not be accepted using any other method.

Your lecturer may use an oral examination or any other mode to test your understanding of the material submitted.

Learning Outcome Covered

1. Use and develop abstract data types.

Marks allocation

Question No.	Marks
1	20
2	25
3	20
4	25
Total	90

Assessment 1 Collections

Question 1 Palindrome

(20 marks)

A Palindrome is word that reads the same backwards as it does forwards. Write a Console application that obtains a word from the user and

using at least one stack , appropriate static variables and the static methods listed below, you are to determine whether the word is a palindrome or not. Then display the appropriate response back to the screen.

Note: Your program must include the following static methods:

- BuildWordStack()
This method receives the word entered by the user as a parameter and has the job of pushing the characters of that

word onto a stack.

- ReverseWord()

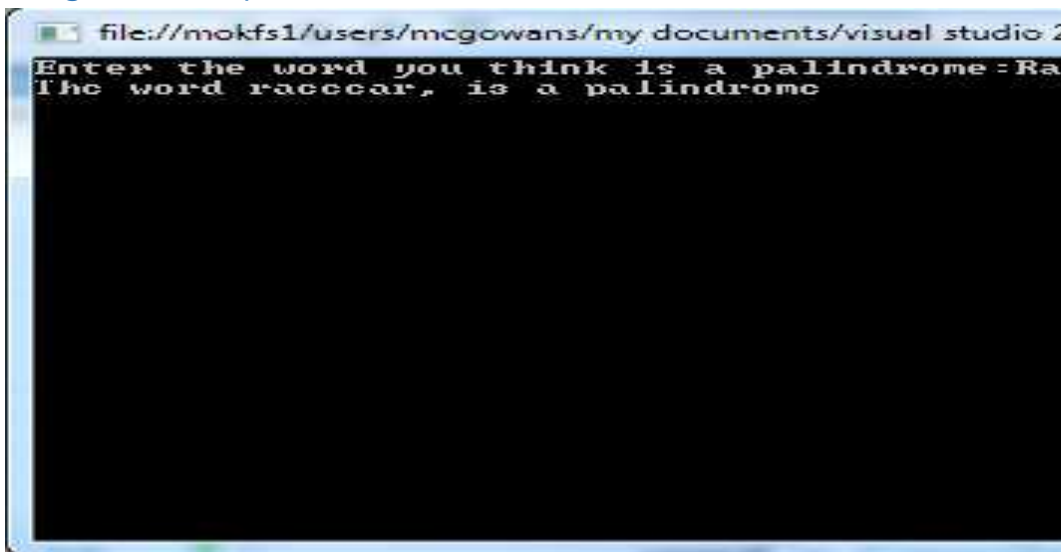
This method receives no parameters but has the job of returning a string; the word (the characters from the stack) in reverse.

- PalindromeCheck()

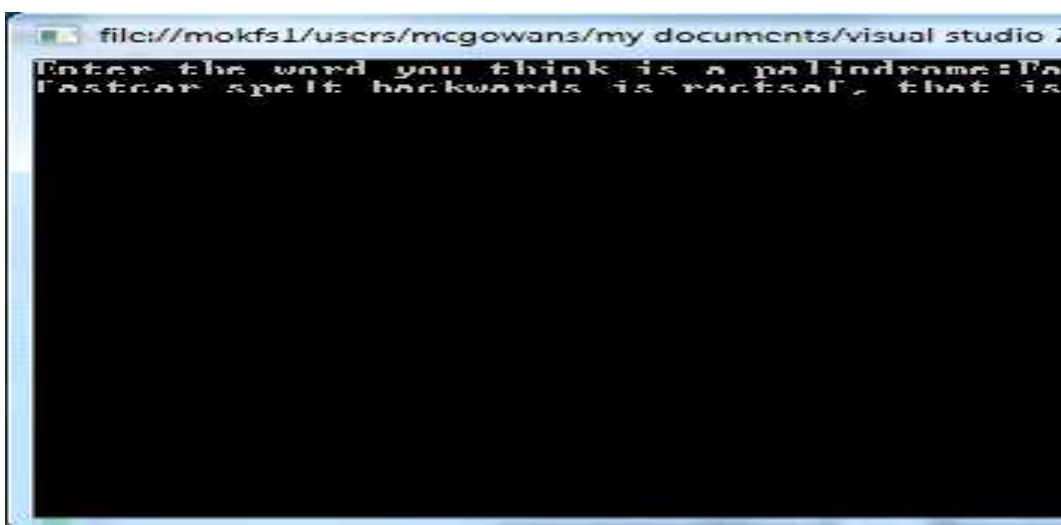
This method receives two parameters (the word entered, and the word in reverse) and has the job of comparing them, and displaying an appropriate message, stating if the word is a palindrome or not.

The program should ignore capitalization and spaces.

Program Samples:



```
file:///mokfs1/users/mcgowans/my documents/visual studio 2
Enter the word you think is a palindrome: Racecar
The word racecar, is a palindrome
```



```
file:///mokfs1/users/mcgowans/my documents/visual studio 2
Enter the word you think is a palindrome: Ractcar
Ractcar spelled backwards is ractcar, that is
```

Marking Criteria

Description	Marks
Adding directive(s)	1
Output prompt/Input of a string	2
Ignore capitalization and spaces	2
BuildWordStack() static method	3
ReverseWord() static method	3
PalindromeCheck() static method	3
Correct utilisation of a stack	2
Displaying appropriate messages to console	2
Comments	2
TOTAL	20

Question 2 Acronym builder

(25 marks)

An **acronym** is a word that is formed from an abbreviated version of a phrase, using the first letters of each Word.

E.g. **ATM** is an acronym for **A**utomatic **T**eller **M**achine.

For this question you are to write a C# Console application to create an acronym from a phrase or sentence given by the user.

You are to do this using two collections:
(with the names outlined in the UML diagram at the bottom of the page)

- An Array of strings to store each of the words within the phrase/sentence.
- A Dictionary to store not only each word (as the value) but also a key (of chars) i.e. the first letter of each word – in uppercase.
This key of capitalised chars should then make up the acronym.

E.g. If the phrase is "Automatic Teller Machine" The Dictionary would look like this:

Key	Value
A	Automatic

T	Teller
M	Machine

This program should contain an Acronym class, with the following attributes and behaviours as outlined in the UML class diagram below:

Acronym
-fullSentence: string -words: string[] -acronymDictionary: Dictionary<char, string>
+ <<constructor>> Acronym(sentence: string) + BuildAcronym() + DisplayAcronym()

In Main your first job is to obtain a sentence from the user. Then use the appropriate public members of the class to do the following:

- The parameterised constructor should receive the sentence as a parameter and then populate the words array.
- In the BuildAcronym method, you are to populate the acronymDictionary with a key (the first letter of each word in uppercase – see validation below for exception here) and a value (the word).
- The DisplayAcronym method, should then display the keys (only) within the dictionary to the screen - showing the final acronym.

VALIDATION: You will need to include validation for those situations where there are two letters in the acronym that are the same, for example; in the phrase laugh out loud. In this acronym LOL, there are two L's. The key in a Dictionary needs to be unique, so in this situation, you should check for any repetition of characters and in the second instance of a repeated character store the key in lowercase. E.G:

Key	Value
L	laugh
O	out
l	loud

The program should display the acronym in upper case – except for the situation mentioned in the Validation above, where the second instance of l is in lowercase.

Marking Criteria

Description	Marks
Adding directive(s)	1
Output prompt/Input a of phrase	2
Acronym class	1
Parameterised constructor	2
BuildAcronym method	4

DisplayAcronym method	4
Acronym in uppercase	1
Validation applied, second instance lowercase	5
Class attributes and behaviours used correctly	3
Comments	2
TOTAL	25

Question 3 Queue Implementation

(20 marks)

A Queue is a commonly known data structure. A queue is similar to a checkout line in a supermarket— the cashier serves the person at the front of the line first. Other customers enter the line only at the back and wait for service. Queue nodes are removed only from the head (or front) of the queue and are inserted only at the tail (or back).

For this reason, a queue is a first-in, first-out (FIFO) data structure.

You are required to Write a C# Console application containing a generic class ("MyQueue") that will have following methods to simulate those of a Queue collection but you are to do this using Array Lists:

Method Name	Description
Enqueue	Adds an object to the end of the Queue
Dequeue	Removes and returns the object at the beginning of the Queue

Remember: Where it says Queue in the description above you are actually dealing with an ArrayList, but to simulate the workings of a Queue.

Marking Criteria

Description	Marks
Adding directive	1
Generic class declaration	2

Enqueue method	6
Dequeue method	6
ArrayList used	3
Comments	2
TOTAL	20

Question 4 Bank Queue

(25 marks)

You are required to Write a C# Console application containing the generic class ("MyQueue") that you created in question 3.

This program is to simulate a queue at the local Westpac Bank. When your program starts you should display a menu of customers for the user to choose from as shown below:

Customer	Processing time
• Tradesman Joe	5 seconds
• Dr Windy Pops	2 seconds
• Grandpa Bob	8 seconds
• Billy the kid	3 seconds
• Chris on crutches	6 seconds

Prompt the user to add 3 customers to the bank queue from the menu given, then show each of them arriving to the front of the queue to see the teller at the appropriate times, which is based around the processing time of the previous customer(s). You will need to research how to implement a wait time of x seconds for this question.

Extra: When Billy the kid gets to see the teller he discovers he has forgotten his ID, so he has to return to his car to get it and then return to the back of the queue to wait again to see the teller, therefore when/if he is chosen from the menu, ensure your program demonstrates this.

Note: You do not have to consider any extra wait time it would take Billy to go out to the car – he simply needs to return to the back of the queue to see the teller again.

Marking Criteria

Description	Marks
Adding directive	1
Includes MyQueue class created in Q2	2
Menu displayed	2
3 customers added to queue, using enqueue method	4
Each of the 3 customers processed, using dequeue method, and correct wait times	9
Code implemented to wait x seconds	3

Extra is implemented	2
Comments	2
TOTAL	25

End -----