

Project 2: Correctness






Due Oct 16 by 11:59pm **Points** 60

No Content

Project 2: Correctness Rubric

Criteria	Ratings			Pts
replaceFirstK: correct performance ✓	2.0 pts Full Marks Works on all inputs.	1.0 pts Good Works on the main cases but fails on one or more boundary cases.	0.0 pts No marks Does not work in a majority of cases.	2.0 pts
replaceFirstK: good code design ✓	2.0 pts Full Marks A reasonable algorithm (even if not correct) for the problem with efficient loops.	1.0 pts Good Reasonable loops/algorithm for the problem but significantly more traversals of the data than necessary.	0.0 pts No Marks The algorithm used is not reasonable for the problem.	2.0 pts
replaceFirstK: efficient memory use ✓	2.0 pts Full Marks Does not create unnecessary copies of a string or string builder. Uses a StringBuilder when creating a string with a loop.	1.0 pts Good Uses StringBuilder correctly anytime a string has to be created with a loop. Creates extra unnecessary strings in solving the problem.	0.0 pts No Marks String creation is required and/or done in a loop and is not using StringBuilder correctly.	2.0 pts
replaceFirstK: uses the proper API ✓	2.0 pts Full Marks Has a loop to traverse the data and does not use any class or method of the API not permitted on the project	0.0 pts No Marks Uses a class or method from the API that is not permitted OR does not have a loop that traverses the data.		2.0 pts
replaceFirstK: break/continue ✓	2.0 pts Full Marks All stopping conditions for the loops (other than method return) is in the loop condition, and no use of break, continue, or code that mimics a break.	0.0 pts No Marks No loops to traverse the data OR a loop is present and it uses break, continue, or code that mimics a break. All loop stopping conditions (other than method return) is not in the loop condition.		2.0 pts
allChars: correct performance ✓	2.0 pts Full Marks Works on all inputs.	1.0 pts Good Works on the main cases but fails on one or more boundary cases.	0.0 pts No Marks Does not work in a majority of cases.	2.0 pts

Criteria	Ratings			Pts
allChars: good code design /	2.0 pts Full Marks A reasonable algorithm (even if not correct) for the problem with efficient loops.	1.0 pts Good Reasonable loops/algorithm for the problem but significantly more traversals of the data than necessary.	0.0 pts No Marks The algorithm used is not reasonable for the problem.	2.0 pts
allChars: efficient memory use /	2.0 pts Full Marks Does not create unnecessary copies of a string or string builder. Uses a StringBuilder when creating a string with a loop.	1.0 pts Good Uses StringBuilder correctly anytime a string has to be created with a loop. Creates extra unnecessary strings in solving the problem.	0.0 pts No Marks String creation is required and/or done in a loop and is not using StringBuilder correctly.	2.0 pts
allChars: uses the proper API /	2.0 pts Full Marks Has a loop to traverse the data and does not use any class or method of the API not permitted on the project	0.0 pts No Marks Uses a class or method from the API that is not permitted OR does not have a loop that traverses the data.		2.0 pts
allChars: break/continue	2.0 pts Full Marks All stopping conditions for the loops (other than method return) is in the loop condition, and no use of break, continue, or code that mimics a break.	0.0 pts No Marks No loops to traverse the data OR a loop is present and it uses break, continue, or code that mimics a break. All loop stopping conditions (other than method return) is not in the loop condition.		2.0 pts
showCharOfString: correct performance /	2.0 pts Full Marks Works on all inputs.	1.0 pts Good Works on the main cases but fails on one or more boundary cases.	0.0 pts No Marks Does not work in a majority of cases.	2.0 pts
showCharOfString: good code design	2.0 pts Full Marks A reasonable algorithm (even if not correct) for the problem with efficient loops.	1.0 pts Good Reasonable loops/algorithm for the problem but significantly more traversals of the data than necessary.	0.0 pts No Marks The algorithm used is not reasonable for the problem.	2.0 pts

Criteria	Ratings			Pts
showCharOfString: efficient memory use 	2.0 pts Full Marks Does not create unnecessary copies of a string or string builder. Uses a StringBuilder when creating a string with a loop.	1.0 pts Good Uses StringBuilder correctly anytime a string has to be created with a loop. Creates extra unnecessary strings in solving the problem.	0.0 pts No Marks String creation is required and/or done in a loop and is not using StringBuilder correctly.	2.0 pts
showCharOfString: uses the proper API 	2.0 pts Full Marks Has a loop to traverse the data and does not use any class or method of the API not permitted on the project	0.0 pts No Marks Uses a class or method from the API that is not permitted OR does not have a loop that traverses the data.		2.0 pts
showCharOfString: break/continue 	2.0 pts Full Marks All stopping conditions for the loops (other than method return) is in the loop condition, and no use of break, continue, or code that mimics a break.	0.0 pts No Marks No loops to traverse the data OR a loop is present and it uses break, continue, or code that mimics a break. All loop stopping conditions (other than method return) is not in the loop condition.		2.0 pts
hangman: correct performance 	2.0 pts Full Marks Works on all inputs.	1.0 pts Good Works on the main cases but fails on one or more boundary cases.	0.0 pts No Marks Does not work in a majority of cases.	2.0 pts
hangman: good code design 	2.0 pts Full Marks A reasonable algorithm (even if not correct) for the problem with efficient loops.	1.0 pts Good Reasonable loops/algorithm for the problem but significantly more traversals of the data than necessary.	0.0 pts No Marks The algorithm used is not reasonable for the problem.	2.0 pts

Criteria	Ratings			Pts
hangman: efficient memory use 	2.0 pts Full Marks Does not create unnecessary copies of a string or string builder. Uses a StringBuilder when creating a string with a loop.	1.0 pts Good Uses StringBuilder correctly anytime a string has to be created with a loop. Creates extra unnecessary strings in solving the problem.	0.0 pts No Marks String creation is required and/or done in a loop and is not using StringBuilder correctly.	2.0 pts
hangman: uses the proper API 	2.0 pts Full Marks Has a loop to traverse the data and does not use any class or method of the API not permitted on the project	0.0 pts No Marks Uses a class or method from the API that is not permitted OR does not have a loop that traverses the data.		2.0 pts
hangman: break/continue 	2.0 pts Full Marks All stopping conditions for the loops (other than method return) is in the loop condition, and no use of break, continue, or code that mimics a break.	0.0 pts No Marks No loops to traverse the data OR a loop is present and it uses break, continue, or code that mimics a break. All loop stopping conditions (other than method return) is not in the loop condition.		2.0 pts
1D hiddenString: correct performance 	2.0 pts Full Marks Works on all inputs.	1.0 pts Good Works on the main cases but fails on one or more boundary cases.	0.0 pts No Marks Does not work in a majority of cases.	2.0 pts
1D hiddenString: good code design 	2.0 pts Full Marks A reasonable algorithm (even if not correct) for the problem with efficient loops.	1.0 pts Good Reasonable loops/algorithm for the problem but significantly more traversals of the data than necessary.	0.0 pts No Marks The algorithm used is not reasonable for the problem.	2.0 pts

Criteria	Ratings			Pts
1D hiddenString: efficient memory use 	2.0 pts Full Marks Does correct array manipulation without creating unnecessary arrays, Strings, StringBuilders or other extra memory usage.	1.0 pts Good Does reasonable array manipulation but creates an unnecessary array, String, StringBuilder or some unnecessary additional memory use.	0.0 pts No Marks Is not doing proper array manipulation OR creates many unnecessary arrays (or other) in solving the problem.	2.0 pts
1D hiddenString: uses the proper API 	2.0 pts Full Marks Has a loop to traverse the data and does not use any class or method of the API not permitted on the project	0.0 pts No Marks Uses a class or method from the API that is not permitted OR does not have a loop that traverses the data.		2.0 pts
1D hiddenString: break/continue 	2.0 pts Full Marks All stopping conditions for the loops (other than method return) is in the loop condition, and no use of break, continue, or code that mimics a break.	0.0 pts No Marks No loops to traverse the data OR a loop is present and it uses break, continue, or code that mimics a break. All loop stopping conditions (other than method return) is not in the loop condition.		2.0 pts
2D hiddenString: correct performance	2.0 pts Full Marks Works on all inputs.	1.0 pts Good Works on the main cases but fails on one or more boundary cases.	0.0 pts No Marks Does not work in a majority of cases.	2.0 pts
2D hiddenString: good code design	2.0 pts Full Marks A reasonable algorithm (even if not correct) for the problem with efficient loops. /	1.0 pts Good Reasonable loops/algorithm for the problem but significantly more traversals of the data than necessary.	0.0 pts No Marks The algorithm used is not reasonable for the problem.	2.0 pts

Criteria	Ratings			Pts
2D hiddenString: efficient memory use	2.0 pts Full Marks Does correct array manipulation without creating unnecessary arrays, Strings, StringBuilders or other extra memory usage.	1.0 pts Good Does reasonable array manipulation but creates an unnecessary array, String, or StringBuilder or some unnecessary additional memory use.	0.0 pts No Marks Is not doing proper array manipulation OR creates many unnecessary arrays (or other) in solving the problem.	2.0 pts
2D hiddenString: uses the proper API	2.0 pts Full Marks Has a loop to traverse the data and does not use any class or method of the API not permitted on the project	0.0 pts No Marks Uses a class or method from the API that is not permitted OR does not have a loop that traverses the data.		2.0 pts
2D hiddenString: break/continue	2.0 pts Full Marks All stopping conditions for the loops (other than method return) is in the loop condition, and no use of break, continue, or code that mimics a break.	0.0 pts No Marks No loops to traverse the data OR a loop is present and it uses break, continue, or code that mimics a break. All loop stopping conditions (other than method return) is not in the loop condition.		2.0 pts
Code Compiles	0.0 pts Code Compiles	-10.0 pts Code Fails To Compile		0.0 pts
Total Points: 60.0				