Strings and Streams

Complete this worksheet by answering the questions using the given program. Make sure to color or **bold** your answers so they are distinguishable from the question.

Getting Started

- ☐ Make a copy of this document by selecting “Make a Copy” from the File menu.
- ☐ Go to https://replit.com/~ and log in with Gmail using your Weber State email address.
- ☐ Open the code linked below
- ☐ Click **fork**, located in the upper right-hand corner, to create your own copy
- ☐ In your copy, go to the upper right-hand corner, and click **invite**. Then, choose **Generate a Join Link**.
- ☐ Paste the generated link in the second bullet point below

Worksheet Program Links

- *Starter Code*: Repl Strings and Streams
- *Finished Program Code*: https://replit.com/join/bvpbfjblmb-trinanixon1

Part 1: c-style strings

In this section, we will observe and practice using c-style strings. We will allow the user to enter a sentence, and we will report the number of spaces, characters, and words in the sentence. We will then create an Acronym out of the first letter in each word.

1. Observe the c-string named sentence. How many characters is it able to contain? (Look at the variable SIZE) 9
2. Observe the use of the strlen function. We will test if the strlen function finds the number of characters or the max length of the string. Run the program, and enter the word “Hello”. How many characters did the strlen return? 5
3. Let's see what happens if we exceed the length of the characters. Run the program again, this time enter something long, such as Mississippi. What gets displayed? Characters: 11 Spaces: 0 Words: 1, but then it crashes and gives the error ***stack smashing detected***: terminated
4. Modify the SIZE variable to store 52 characters instead of 9.

5. Run the program again. This time enter “Hello World” or something with multiple words. Do multiple words get displayed in the sentence? No, it still says Words: 1.
6. The getline command is used to get read a line instead of just a word. Comment out the cin statement and remove the comment from the getline statement. Run the program again and enter a sentence. Does the program read the entire sentence this time? Yes, says Words: 2.
7. What do you think is the difference between the getline statement and the cin>> statement? Getline reads an entire line of text until the Enter key is pressed, while cin reads until it encounters a space or newline. Cin is good for reading individual words or numbers, while getline is good for reading entire lines of text.
8. Observe the first for-loop in the function. It is counting the number of spaces in the c-string. What syntax is used to isolate an individual character of the c-string? sentence[i] is used to isolate an individual character of the c-string.
9. In the second for-loop, we are going to create an acronym out of the first letter of each word. For example, given the sentence *random access memory*, the acronym would be RAM. Observe the variable acronym. How many characters can it store?32
10. What is stored in character 0 of the acronym? Will it be uppercase or lowercase? What function is being used to change the case? The lowercase version of the first character of ‘sentence’. tolower() is the function being used to change the case
11. Change the function to make it upper case.
12. Observe the for-loop. It is iterating through the sentence. What is being used to determine the start of a new word? A space character (‘ ‘)
13. Observe the position of the characters below in the following two strings. Are the index positions for the first characters in each word in the same index value as the same letters in the acronym? No.

char sentence[SIZE]

r	a	n	d	o	m		a	c	c	e	s	s		m	e	m	o	r	y
0	1	2	3	4	5	6	7	8	9	0	11	12	13	14	15	16	17	18	19

char acronym[31]

R	A	M	\0
0	1	2	3

14. Observe the for loop under the Create Acronym comment.
    a. What is the name of the variable in the for loop that is iterating over the string sentence? i is the variable that is iterating over the string sentence
    b. What is the name of the variable used to iterate through the acronym? acronymPos
15. There is a logic error with the acronym program. Currently in the if statement, is checking for a space, this means that position i of the sentence is at the space instead of the first character in the next word. Modify the line of code to set the acronym to the next letter in the string instead of the space. Hint: add 1 to i..
16. Add the function to make it upper case.
17. Test the program to see if it works.
18. Scroll down to the main() function at the bottom of the file. Add a comment in front of Example1() and remove the comment for Example2(). Then continue on to the next section.

Part 2: std::strings

The following example will demonstrate issues with the getline and >> styles of input.

1. Look at each of the variables being input, a word, a sentence, and a number. What type of input statement is being used for each, getline() or cin statement?
    a. Word - cin

b. Sentence - cin

c. Num - cin

2. Run the program, when prompted for a word, enter a word. When prompted for a sentence, still enter just a single word. When prompted for a number, enter a number. Did the program work as intended? Yes.
3. Run the program again, when prompted, enter a word. When prompted for a sentence, enter multiple words. Does the program allow you to enter a number? Why or why not? No, it doesn't allow me to enter a number. It's because cin is being used so it reads input until a space is encountered, so when a sentence with multiple words is entered it reads the first word and the rest of the words are left in memory. When the program asks for a number, it reads the remaining words as part of the number input.
4. Observe the cin statement for the sentence. This only allows you to enter a single word. Comment out the cin statement and remove the comment from the getline statement.
5. Run the program again and enter a word a word. Does the program allow you to enter a sentence? Why or why not? No, because when you type a word and press Enter, the word is stored in the 'word' variable but the 'Enter' is stored in memory, so when getline(cin,sentence) is executed, it also reads the remaining newline character from the memory. It reads it as an empty line and moves to the next input asking for a number.
6. A cin statement disregards endl characters whereas getline reads until it finds an endl character. This means that if you have a cin statement followed by a getline statement, you must ignore the endl character in the stream. Remove the comment from the cin.ignore() statement. Then run the program again. Does it work as expected? Yes.
7. Scroll down to the main() function at the bottom of the file. Add a comment in front of Example2() and remove the comment for Example3(). Then continue on to the next section.

Part 3: StringStream

Stringstream may be used to parse or construct streams. Observe the stringstream in action.

1. Look at the #include statements at the top of the page. Which #include statement is used for the stringstream? #include <sstream>
2. Look back at the code for Example3(). What separates each word in the variable *parse*? Dashes (-);
3. Observe the stringstream. What is the name given to the stringstream object? What parameter is being passed to the stringstream object? sParse is the name given to the object and the parameter is the string 'parse'.
4. The getline function is used in conjunction with the stringstream object to parse strings. How does it know the character that is being used to parse the string? The 3rd parameter passed is the delimiter - so in this case it's a '-'
5. How does the program know when there are no more words to parse? the getline function is looking for dashes and when it reaches the end and there are no more dashes , it will return false and the loop will terminate.
6. Modify the variable parse to separate the words with a slash/ instead of a dash-.
7. Modify the while loop so it still continues to work with a slash.
8. Stringsteam can also be used to concatenate variables together into a string. What is the name of the stringstream object used to concatenate? Are there any parameters this time? Why or why not? The name of the stringstream object is sConcat; no parameter. Because we want it empty initially, and then we add to it to build a string of words.
9. What operator is used to add to the stringstream object? The stream insertion operator "<<"
10. What function is used to convert the stringstream into a string? sConcat.str()

11. Scroll down to the main() function at the bottom of the file. Add a comment in front of Example3() and remove the comment for Example4(). Then continue on to the next section.

Part 4: String Logic - Pig Latin

We are going to use what we know about strings to write a simple program to convert a sentence into Pig Latin.

Pig Latin is a language game where you take the first syllable of each word and move it to the end of the word and add “ay”. So water would be aterway cat would be atcay. In the case that a word begins in a vowel, then just “yay” is added to the end, so apple becomes appleyay.

1. Observe the function findVowelPos(). What is contained in the vowels variable? “aeiou”
2. What do you think the transform() function is doing to the string? Why do you think it might need to be in a certain case? transform() will convert all the characters in the string to lowercase. Because the vowels string is “aeiou”, and to be able to find all the vowels in any given string, it should also be lowercase as well.
3. The outer for-loop is iterating the word. The inner for loop is iterating through each of the vowels. Modify the if statement to use the .at() function or the [ ] to compare the character at position *i* in the word, to the character in position *j* of the vowels.
4. Now look at the code in the Example4() function. What type of input is used for the sentence? There is not another cin statement, does there need to be an ignore statement()? ‘getline(cin, sentence)’ is the type of input used for sentence. cin.ignore() is used when switching from cin >> to getline() to read input, but since there is not another cin statement, it’s not necessary.
5. Observe the declaration of the stringstream. Are there any parameters? Look at the command after the getline for the sentence. How is the string sent to the stringstream?

No, there are no parameters for the stringstream ss; after getting the input for sentence from the user using ‘getline(cin, sentence)’ the ‘sentence’ variable is sent to the stringstream ‘ss’ using the << operator, which inserts the data into the stringstream.

6. Observe the function call to findVowelPos(). What variable stores the vowel position?

   vowelPos

7. Observe the if statement in the while loop. If there are no vowels, “yay” is added to the string. If there is a vowel, then we need to put the first part syllable at the end of the word. Use the substr function with the vowelPos variable to define the word. Set the word equal to a substring of the word from the vowel to the end plus a substring of the start of the word to the vowel, plus “ay”.
8. Test the program to make sure it works.