[ZANG] - [Chuanjie] - LNXOS Assignment

Student ID: 699382953

[chuanjiezang@gmail.com](mailto:chuanjiezang@gmail.com)

Task 1

Write a Python function that accepts a sentence composed of words from a user and displays the following:

a) Middle word

b) Longest word in the sentence

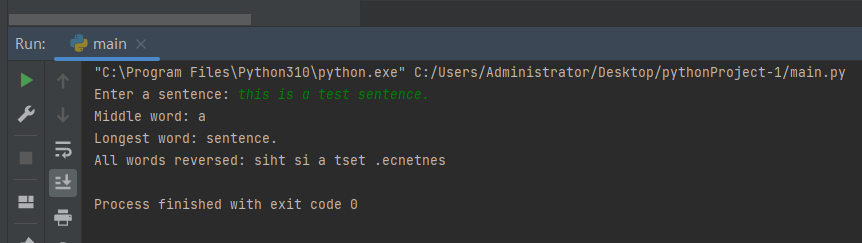
c) All the words in sentence reversed

You can take any string as input from a user and perform the above operation.

Here is the code:

# Assignment 2 - task 1  
def analyze\_sentence(sentence):  
 # Split the sentence into words  
 words = sentence.split()  
  
 # Find the middle word  
 middle\_index = len(words) // 2  
 middle\_word = words[middle\_index]  
  
 # Find the longest word  
 longest\_word = max(words, key=len)  
  
 # Reverse all the words in the sentence  
 reversed\_words = [word[::-1] for word in words]  
  
 return middle\_word, longest\_word, reversed\_words  
  
# Example usage:  
user\_sentence = input("Enter a sentence: ")  
middle\_word, longest\_word, reversed\_words = analyze\_sentence(user\_sentence)  
  
print("Middle word:", middle\_word)  
print("Longest word:", longest\_word)  
print("All words reversed:", ' '.join(reversed\_words))





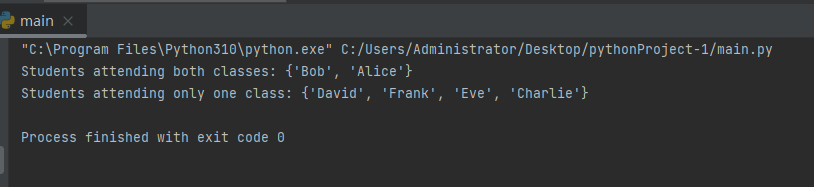
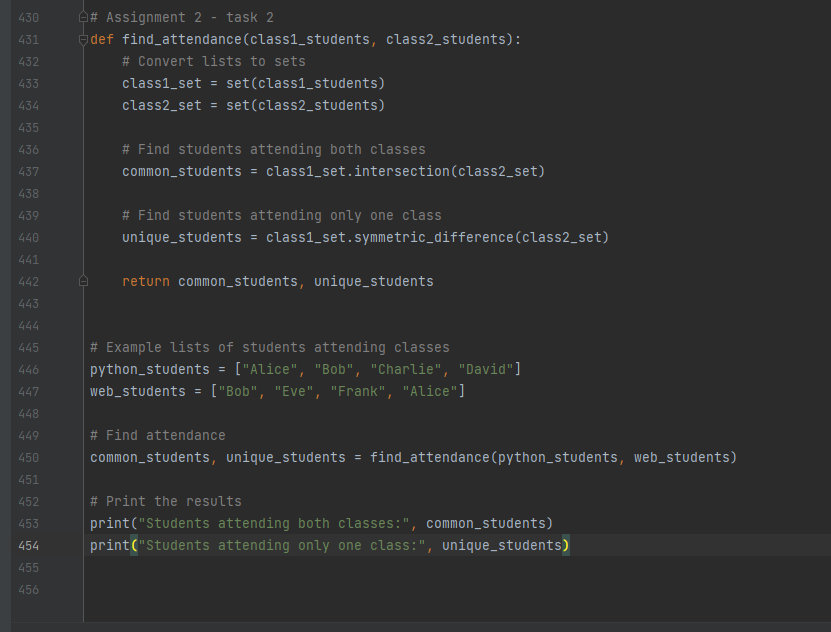
Task 2:

Consider the following scenario. You have a list of students who are attending class “Python” and another list of students who are attending class “Web Application”. Find the list of students who are attending both classes. Also find the list of students who are not common in both the classes.

Print it. You can take any two lists of string as input from the user and perform the above operation, or you can declare any two lists in the program and perform the operation.

Here is the code:

# Assignment 2 - task 2  
def find\_attendance(class1\_students, class2\_students):  
 # Convert lists to sets  
 class1\_set = set(class1\_students)  
 class2\_set = set(class2\_students)  
  
 # Find students attending both classes  
 common\_students = class1\_set.intersection(class2\_set)  
  
 # Find students attending only one class  
 unique\_students = class1\_set.symmetric\_difference(class2\_set)  
  
 return common\_students, unique\_students  
  
  
# Example lists of students attending classes  
python\_students = ["Alice", "Bob", "Charlie", "David"]  
web\_students = ["Bob", "Eve", "Frank", "Alice"]  
  
# Find attendance  
common\_students, unique\_students = find\_attendance(python\_students, web\_students)  
  
# Print the results  
print("Students attending both classes:", common\_students)  
print("Students attending only one class:", unique\_students)



Task 3:

Complete the programs outline below.

a. Addition of two numbers using Perl. You can use any two numbers as input from a user and perform the above operation.

b. Write the Perl script to Remove Empty Directories. You can take any string as input from a user and perform the above operation.

Here is the code:

1. Addition of two numbers using Perl:

Use:  
chmod +x task3-a.pl  
before execution

#!/usr/bin/perl

# Prompt the user to enter two numbers

print "Enter the first number: ";

$num1 = <STDIN>;

chomp($num1);

print "Enter the second number: ";

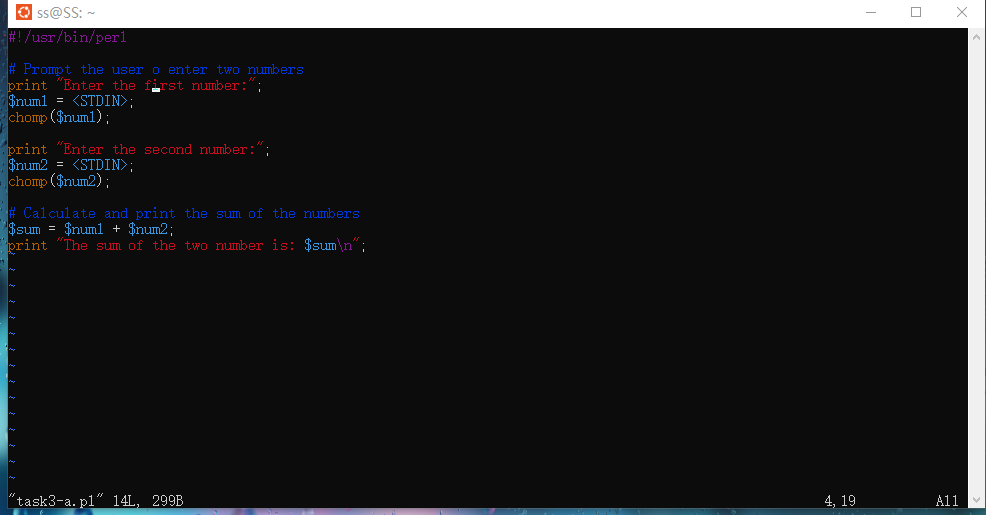
$num2 = <STDIN>;

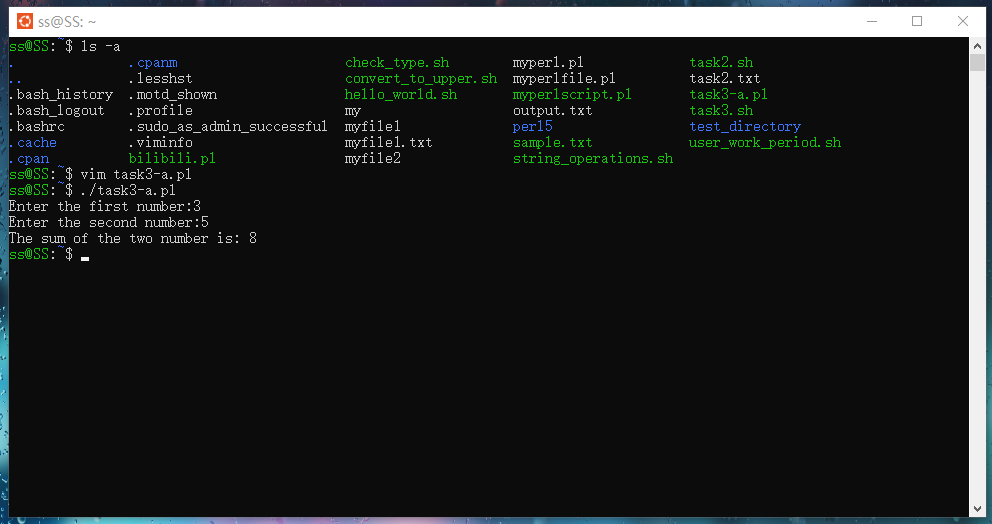
chomp($num2);

# Calculate and print the sum of the two numbers

$sum = $num1 + $num2;

print "The sum of the two numbers is: $sum\n";





b. Perl script to Remove Empty Directories:

use strict;

use warnings;

use File::Find;

Use:  
chmod +x task3-a.pl  
before execution

#!/usr/bin/perl

use strict;

use warnings;

use File::Find;

# Prompt the user to enter the directory path

print "Please enter the directory path to check: ";

my $dir\_path = <STDIN>;

chomp($dir\_path);

# Check and remove empty directories

find(sub {

if (-d && rmdir($\_)) {

print "Empty directory removed: $File::Find::name\n";

}

}, $dir\_path);

