**Assignment for CESC Python Coursework**

**Exercise 1:** Print “helloworld”.

**Exercise 2:** Write a program that uses input to prompt a user for their name and then welcomes them.

Enter your name: Chuck

Hello Chuck

**Exercise 3:** Write a program to prompt the user for hours and rate per hour to compute gross pay.

Enter Hours: 35

Enter Rate: 2.75

Pay: 96.25

We won't worry about making sure our pay has exactly two digits after the decimal place for now. If you want, you can play with the built-in Python round function to properly round the resulting pay to two decimal places.

**Exercise 4:** Assume that we execute the following assignment statements:

width = 17

height = 12.0

For each of the following expressions, write the value of the expression and the type (of the value of the expression).

width//2

width/2.0

height/3

1 + 2 \* 5

Use the Python interpreter to check your answers.

**Exercise 5:** Write a program which prompts the user for a Celsius temperature, convert the temperature to Fahrenheit, and print out the converted temperature.

**Exercise 6**: Rewrite your pay computation to give the employee 1.5 times the hourly rate for hours worked above 40 hours.

Enter Hours: 45

Enter Rate: 10

Pay: 475.0

**Exercise 7**: Rewrite your pay program using try and except so that your program handles non-numeric input gracefully by printing a message and exiting the program. The following shows two executions of the program:

Enter Hours: 20

Enter Rate: nine

Error, please enter numeric input

Enter Hours: forty

Error, please enter numeric input

Exercise 8: Write a program to prompt for a score between 0.0 and 1.0. If the score is out of range, print an error message. If the score is between 0.0 and 1.0, print a grade using the following table:

Score Grade

>= 0.9 A

>= 0.8 B

>= 0.7 C

>= 0.6 D

< 0.6 F

Enter score: 0.95

A

Enter score: perfect

Bad score

Enter score: 10.0

Bad score

Enter score: 0.75

C

Enter score: 0.5

F

Run the program repeatedly as shown above to test the various different values for input.

Exercise 9: What is the purpose of the "def" keyword in Python?

Exercise 10: What will the following Python program print out?

def fred():

print("Zap")

def jane():

print("ABC")

jane()

fred()

jane()

a) Zap ABC jane fred jane

b) Zap ABC Zap

c) ABC Zap jane

d) ABC Zap ABC

e) Zap Zap Zap

**Exercise 11**: Rewrite your pay computation with time-and-a-half for overtime and create a function called computepay which takes two parameters (hours and rate).

Enter Hours: 45

Enter Rate: 10

Pay: 475.0

**Exercise 12**: Rewrite the grade program from the previous chapter using a function called computegrade that takes a score as its parameter and returns a grade as a string.

Score Grade

>= 0.9 A

>= 0.8 B

>= 0.7 C

>= 0.6 D

< 0.6 F

Enter score: 0.95

A

Enter score: perfect

Bad score

Enter score: 10.0

Bad score

Enter score: 0.75

C

Enter score: 0.5

F

Run the program repeatedly to test the various different values for input

**Exercise 13.** Write a Python program to compute and print sum of two given integers (more than or equal to zero). If given integers or the sum have more than 80 digits, print "overflow".   
Input first integer:  
25  
Input second integer:  
22  
Sum of the two integers: 47

**Exercise 14.** Write a Python program that accepts six numbers as input and sorts them in descending order.   
**Input:**  
Input consists of six numbers n1, n2, n3, n4, n5, n6 (-100000 <= n1, n2, n3, n4, n5, n6 <= 100000). The six numbers are separated by a space.  
Input six integers:  
15 30 25 14 35 40  
After sorting the said integers:  
40 35 30 25 15 14

**Exercise 14.** Write a Python program to test whether two lines PQ and RS are parallel. The four points are P(x1, y1), Q(x2, y2), R(x3, y3), S(x4, y4).   
**Input:**  
x1,y1,x2,y2,x3,y3,xp,yp separated by a single space  
Input x1,y1,x2,y2,x3,y3,xp,yp:  
2 5 6 4 8 3 9 7  
PQ and RS are not parallel

**Exercise 15.** Write a Python program to find the maximum sum of a contiguous subsequence from a given sequence of numbers a1, a2, a3, ... an. A subsequence of one element is also a continuous subsequence.   
**Input:**  
You can assume that 1 <= n <= 5000 and -100000 <= ai <= 100000.  
Input numbers are separated by a space.  
Input 0 to exit.  
Input number of sequence of numbers you want to input (0 to exit):  
3  
Input numbers:  
2  
4  
6  
Maximum sum of the said contiguous subsequence: 12  
Input number of sequence of numbers you want to input (0 to exit):  
0

**Exercise 16.** There are two circles C1 with radius r1, central coordinate (x1, y1) and C2 with radius r2 and central coordinate (x2, y2).

Write a Python program to test the followings -

* "C2 is in C1" if C2 is in C1
* "C1 is in C2" if C1 is in C2
* "Circumference of C1 and C2 intersect" if circumference of C1 and C2 intersect, and
* "C1 and C2 do not overlap" if C1 and C2 do not overlap.

**Input:**  
Input numbers (real numbers) are separated by a space.  
Input x1, y1, r1, x2, y2, r2:  
5 6 4 8 7 9  
C1 is in C2

**Exercise 17.** Write a Python program to that reads a date (from 2016/1/1 to 2016/12/31) and prints the day of the date. Jan. 1, 2016, is Friday. Note that 2016 is a leap year.   
**Input:**  
Two integers m and d separated by a single space in a line, m ,d represent the month and the day.  
Input month and date (separated by a single space):  
5 15  
Name of the date: Sunday

**Exercise 18.** Write a Python program which reads a text (only alphabetical characters and spaces.) and prints two words. The first one is the word which is arise most frequently in the text. The second one is the word which has the maximum number of letters.

Note: A word is a sequence of letters which is separated by the spaces.

**Input:**  
A text is given in a line with following condition:  
a. The number of letters in the text is less than or equal to 1000.  
b. The number of letters in a word is less than or equal to 32.  
c. There is only one word which is arise most frequently in given text.  
d. There is only one word which has the maximum number of letters in given text.  
Input text: Thank you for your comment and your participation.  
Output: your participation.

**Exercise 19.** Write a Python program that reads n digits (given) chosen from 0 to 9 and prints the number of combinations where the sum of the digits equals to another given number (s). Do not use the same digits in a combination.   
**Input:**  
Two integers as number of combinations and their sum by a single space in a line. Input 0 0 to exit.  
Input number of combinations and sum, input 0 0 to exit:  
5 6  
2 4  
0 0  
2

**Exercise 20.** Write a Python program which reads the two adjoined sides and the diagonal of a parallelogram and check whether the parallelogram is a rectangle or a rhombus.   
According to Wikipedia- parallelograms: In Euclidean geometry, a parallelogram is a simple (non-self-intersecting) quadrilateral with two pairs of parallel sides. The opposite or facing sides of a parallelogram are of equal length and the opposite angles of a parallelogram are of equal measure.  
rectangles: In Euclidean plane geometry, a rectangle is a quadrilateral with four right angles. It can also be defined as an equiangular quadrilateral, since equiangular means that all of its angles are equal (360°/4 = 90°). It can also be defined as a parallelogram containing a right angle.  
rhombus: In plane Euclidean geometry, a rhombus (plural rhombi or rhombuses) is a simple (non-self-intersecting) quadrilateral whose four sides all have the same length. Another name is equilateral quadrilateral, since equilateral means that all of its sides are equal in length. The rhombus is often called a diamond, after the diamonds suit in playing cards which resembles the projection of an octahedral diamond, or a lozenge, though the former sometimes refers specifically to a rhombus with a 60° angle, and the latter sometimes refers specifically to a rhombus with a 45° angle.  
Input:  
Two adjoined sides and the diagonal.  
1 <= ai, bi, ci <= 1000, ai + bi > ci  
Input two adjoined sides and the diagonal of a parallelogram (comma separated):  
3,4,5  
This is a rectangle.

**Exercise 21.** Write a Python program to compute the sum of first n given prime numbers.   
**Input:**  
n ( n <= 10000). Input 0 to exit the program.  
Input a number (n<=10000) to compute the sum:(0 to exit)  
25  
Sum of first 25 prime numbers:  
1060

**Exercise 22.**Write a Python program to find the index of an item in a specified list.   
**Exercise** **23.**Write a Python program to flatten a shallow list.   
**Exercise** **24.**Write a Python program to append a list to the second list.   
**Exercise** **25.**Write a Python program to select an item randomly from a list.   
**Exercise** **26.**Write a python program to check whether two lists are circularly identical.   
**Exercise** **27.**Write a Python program to find the second smallest number in a list.   
**28.**Write a Python program to find the second largest number in a list.   
**Exercise** **29.**Write a Python program to get unique values from a list.   
**Exercise** **30.**Write a Python program to get the frequency of the elements in a list.   
**Exercise** **31.**Write a Python program to count the number of elements in a list within a specified range.   
**Exercise** **32.** Write a Python program to print a dictionary line by line. 

**Exercise** **33.** Write a Python program to check multiple keys exists in a dictionary. 

**Exercise** **34.** Write a Python program to count number of items in a dictionary value that is a list. 

**Exercise** **35.** Write a Python program to sort Counter by value.   
Sample data : {'Math':81, 'Physics':83, 'Chemistry':87}  
Expected data: [('Chemistry', 87), ('Physics', 83), ('Math', 81)]

**Exercise** **36.** Write a Python program to create a dictionary from two lists without losing duplicate values.   
Sample lists: ['Class-V', 'Class-VI', 'Class-VII', 'Class-VIII'], [1, 2, 2, 3]  
Expected Output: defaultdict(<class 'set'>, {'Class-V': {1}, 'Class-VI': {2}, 'Class-VII': {2}, 'Class-VIII': {3}})

**Exercise** **37.** Write a Python program to replace dictionary values with their average. 

**Exercise** **38.** Write a Python program to match key values in two dictionaries.   
Sample dictionary: {'key1': 1, 'key2': 3, 'key3': 2}, {'key1': 1, 'key2': 2}  
Expected output: key1: 1 is present in both x and y

**Exercise 39.** Write a Python program to convert a tuple of string values to a tuple of integer values.   
Original tuple values:  
(('333', '33'), ('1416', '55'))  
New tuple values:  
((333, 33), (1416, 55))

**Exercise 40.** Write a Python program to convert a given tuple of positive integers into an integer.   
Original tuple:  
(1, 2, 3)  
Convert the said tuple of positive integers into an integer:  
123  
Original tuple:  
(10, 20, 40, 5, 70)  
Convert the said tuple of positive integers into an integer:  
102040570

**Exercise 41.** Write a Python program to check if a specified element presents in a tuple of tuples.   
Original list:  
(('Red', 'White', 'Blue'), ('Green', 'Pink', 'Purple'), ('Orange', 'Yellow', 'Lime'))  
Check if White presenet in said tuple of tuples!  
True  
Check if White presenet in said tuple of tuples!  
True  
Check if Olive presenet in said tuple of tuples!  
False

**Exercise 42.** Write a Python program to compute element-wise sum of given tuples.   
Original lists:  
(1, 2, 3, 4)  
(3, 5, 2, 1)  
(2, 2, 3, 1)  
Element-wise sum of the said tuples:  
(6, 9, 8, 6)

**Exercise 43.** Write a Python program to compute the sum of all the elements of each tuple stored inside a list of tuples.   
Original list of tuples:  
[(1, 2), (2, 3), (3, 4)]  
Sum of all the elements of each tuple stored inside the said list of tuples:  
[3, 5, 7]  
Original list of tuples:  
[(1, 2, 6), (2, 3, -6), (3, 4), (2, 2, 2, 2)]  
Sum of all the elements of each tuple stored inside the said list of tuples:  
[9, -1, 7, 8]

**Exercise 44.** Write a Python program to find maximum and the minimum value in a set.

**Exercise 45.** Write a Python program to find the length of a set.

**Exercise 46.** Write a Python program to check if a given value is present in a set or not.

**Exercise 47.** Write a Python program to check if two given sets have no elements in common.

**Exercise 48.** Write a Python program to check if a given set is superset of itself and superset of another given set.

**Exercise 49.** Write a Python program to find the elements in a given set that are not in another set.

**Exercise 50.** Write a Python program to check a given set has no elements in common with other given set.

**Exercise 51.**Write a Python program to construct the following pattern, using a nested for loop.

\*   
\* \*   
\* \* \*   
\* \* \* \*   
\* \* \* \* \*   
\* \* \* \*   
\* \* \*   
\* \*   
\*

**Exercise 52.** Write a Python program that accepts a word from the user and reverse it. 

**Exercise 53.** Write a Python program to count the number of even and odd numbers from a series of numbers.   
*Sample numbers* : numbers = (1, 2, 3, 4, 5, 6, 7, 8, 9)   
*Expected Output* :  
Number of even numbers : 5  
Number of odd numbers : 4

**Exercise 54.** Write a Python program that prints each item and its corresponding type from the following list.  
*Sample List* : datalist = [1452, 11.23, 1+2j, True, 'w3resource', (0, -1), [5, 12], {"class":'V', "section":'A'}]

**Exercise 55.** Write a Python program that prints all the numbers from 0 to 6 except 3 and 6.  
Note : Use 'continue' statement.  
Expected Output : 0 1 2 4 5

**Exercise 56.** Write a Python program to get the Fibonacci series between 0 to 50.   
Note : The Fibonacci Sequence is the series of numbers :  
0, 1, 1, 2, 3, 5, 8, 13, 21, ....  
Every next number is found by adding up the two numbers before it.  
Expected Output : 1 1 2 3 5 8 13 21 34

**Exercise 57.** Write a Python program which iterates the integers from 1 to 50. For multiples of three print "Fizz" instead of the number and for the multiples of five print "Buzz". For numbers which are multiples of both three and five print "FizzBuzz".  
*Sample Output* :  
fizzbuzz  
1  
2  
fizz  
4  
buzz

**Exercise 58.**Write a Python program which takes two digits m (row) and n (column) as input and generates a two-dimensional array. The element value in the i-th row and j-th column of the array should be i\*j.   
Note :  
i = 0,1.., m-1  
j = 0,1, n-1.

Test Data : Rows = 3, Columns = 4  
Expected Result : [[0, 0, 0, 0], [0, 1, 2, 3], [0, 2, 4, 6]]

**Exercise 59.**Write a Python program that accepts a sequence of lines (blank line to terminate) as input and prints the lines as output (all characters in lower case). 

**Exercise 60.**Write a Python program which accepts a sequence of comma separated 4 digit binary numbers as its input and print the numbers that are divisible by 5 in a comma separated sequence.   
Sample Data : 0100,0011,1010,1001,1100,1001  
Expected Output : 1010

**Exercise 61.**Write a Python program that accepts a string and calculate the number of digits and letters.   
Sample Data : Python 3.2  
Expected Output :  
Letters 6  
Digits 2

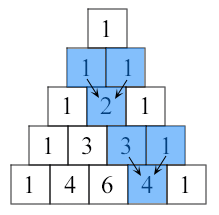
**Exercise 62.**Write a Python program to check the validity of password input by users.   
Validation :

* At least 1 letter between [a-z] and 1 letter between [A-Z].
* At least 1 number between [0-9].
* At least 1 character from [$#@].
* Minimum length 6 characters.
* Maximum length 16 characters.

**Exercise 63.**Write a Python program to find numbers between 100 and 400 (both included) where each digit of a number is an even number. The numbers obtained should be printed in a comma-separated sequence.

**Exercise 64.** Write a Python function that prints out the first n rows of Pascal's triangle.   
Note : Pascal's triangle is an arithmetic and geometric figure first imagined by Blaise Pascal.

Sample Pascal's triangle :



Each number is the two numbers above it added together 

**Exercise 65.** Write a Python function to check whether a string is a pangram or not.   
Note : Pangrams are words or sentences containing every letter of the alphabet at least once.  
For example : "The quick brown fox jumps over the lazy dog"

**Exercise 66.** Write a Python program that accepts a hyphen-separated sequence of words as input and prints the words in a hyphen-separated sequence after sorting them alphabetically.   
*Sample Items*: green-red-yellow-black-white *Expected Result*: black-green-red-white-yellow

**Exercise 67.** Write a Python function to create and print a list where the values are square of numbers between 1 and 30 (both included). 

**Exercise 68.** Write a Python program to make a chain of function decorators (bold, italic, underline etc.) in Python. 

**Exercise 69.** Write a Python program to execute a string containing Python code. 

**Exercise 70.** Write a Python program to access a function inside a function. 

**Exercise 71.** Write a Python program to detect the number of local variables declared in a function. 

**Exercise 72.** Write a Python class to convert an integer to a roman numeral.

**Exercise 73.** Write a Python class to convert a roman numeral to an integer.

**Exercise 74.** Write a Python class to find validity of a string of parentheses, '(', ')', '{', '}', '[' and ']. These brackets must be close in the correct order, for example "()" and "()[]{}" are valid but "[)", "({[)]" and "{{{" are invalid.

**Exercise 75.** Write a Python class to get all possible unique subsets from a set of distinct integers.   
Input : [4, 5, 6]  
Output : [[], [6], [5], [5, 6], [4], [4, 6], [4, 5], [4, 5, 6]]

**Exercise 76.** Write a Python class to find a pair of elements (indices of the two numbers) from a given array whose sum equals a specific target number.   
Input: numbers= [10,20,10,40,50,60,70], target=50  
Output: 3, 4

**Exercise 77.** Write a Python class to find the three elements that sum to zero from a set of n real numbers.   
Input array : [-25, -10, -7, -3, 2, 4, 8, 10]  
Output : [[-10, 2, 8], [-7, -3, 10]]

**Exercise 78.** Write a Python class to implement pow(x, n).

**Exercise 79.** Write a Python class to reverse a string word by word.   
Input string : 'hello .py'  
Expected Output : '.py hello'

**Exercise 81.** Write a Python class which has two methods get\_String and print\_String. get\_String accept a string from the user and print\_String print the string in upper case. 

**Exercise 82 .**Write a Python class named Rectangle constructed by a length and width and a method which will compute the area of a rectangle.   
**Exercise 83.**Write a Python class named Circle constructed by a radius and two methods which will compute the area and the perimeter of a circle.   
**Exercise 84.**Write a Python program to get the class name of an instance in Python. 

**Exercise 85.** Write a Python program to write a list to a file. 

**Exercise 86.** Write a Python program to copy the contents of a file to another file . 

**Exercise 87.** Write a Python program to combine each line from first file with the corresponding line in second file. 

**Exercise 88.** Write a Python program to read a random line from a file. 

**Exercise 89.** Write a Python program to assess if a file is closed or not. 

**Exercise 90.** Write a Python program to remove newline characters from a file. 

**Exercise 91.** Write a Python program that takes a text file as input and returns the number of words of a given text file.   
Note: Some words can be separated by a comma with no space.

**Exercise 92.** Write a Python program to extract characters from various text files and puts them into a list. 

**Exercise 93.** Write a Python program to generate 26 text files named A.txt, B.txt, and so on up to Z.txt. 

**Exercise 94.** Write a Python program to create a file where all letters of English alphabet are listed by specified number of letters on each line. 