# Lab 2: Strings, Lists, and Formatted Output

Name: _____ University Number: _____

## Exercise 1: Operations on a Line of Text via String Manipulation

AIM:

Write a Python program that prompts the user for one line of text without punctuation marks and then asks the user to choose one of the following operations to be done on the line of text:

(a) printing the line of text with all its blank spaces removed,

(b) printing the line of text with each word in reversed order,

(c) printing the number of words in the line of text,

by using string manipulation. Your program should check if the user input of the choice of operation is valid. You can assume that the entered line of text has no punctuation marks. Here are the sample input and output of this program:

```
Enter a line of text without punctuation mark:
I am very very   happy
Type a number 1 to 3 to do one of the following operations
 1 - print the line of text with all its blank spaces removed
 2 - print the line of text with each word in reversed order
 3 - print the number of words in the line of text
Enter your choice (1-3): 1
The line of text with all its blank spaces removed:
Iamveryveryhappy
Enter a line of text without punctuation mark:
I am very very   happy
Type a number 1 to 3 to do one of the following operations
 1 - print the line of text with all its blank spaces removed
 2 - print the line of text with each word in reversed order
 3 - print the number of words in the line of text
Enter your choice (1-3): 2
The line of text with each word in reversed order:
I ma yrev yrev   yppah
Enter a line of text without punctuation mark:
```

```
I am very very   happy
Type a number 1 to 3 to do one of the following operations
 1 - print the line of text with all its blank spaces removed
 2 - print the line of text with each word in reversed order
 3 - print the number of words in the line of text
Enter your choice (1-3): 3
The number of words in the line of text: 5
Enter a line of text without punctuation mark:
I am very very   happy
Type a number 1 to 3 to do one of the following operations
 1 - print the line of text with all its blank spaces removed
 2 - print the line of text with each word in reversed order
 3 - print the number of words in the line of text
Enter your choice (1-3): 5
Invalid input. You must type 1 or 2 or 3!
```

## ALGORITHM:

1. Start
2. Input a line of text without punctuation mark
3. Ask the user to choose one of the following operations to be done on the line of text:
   (a) Printing the line of text with all its blank spaces removed (by using the string methods `split` and `join`)
   (b) Printing the line of text with each word in reversed order (by using the string method `split` to obtain a list of strings composed of each word in the text and then using the slice operator `:` to slice each string in this list properly)
   (c) Printing the number of words in the line of text (by using the string method `split` to obtain a list of strings composed of each word in the text and then counting the number of elements in this list)

   If the user choice of operation is valid, then perform the chosen operation.
4. Output the result of the operation if the user choice of operation is valid; otherwise output an error message
5. End

(The program is shown on the next page.)

PROGRAM:

```
# Exercise 1: Operations on a Line of Text via String Manipulation
# Written by F K Chow, HKU
# Last update: 2022/1/27

# Input a line of text without punctuation mark
string = input('Enter a line of text without punctuation mark:\n')

# Ask the user to choose the operation to be done on the line of text
# and perform the chosen operation only if the user choice is valid
print('Type a number 1 to 3 to do one of the following operations')
print(' 1 - print the line of text with all its blank spaces removed')
print(' 2 - print the line of text with each word in reversed order')
print(' 3 - print the number of words in the line of text')
choice = int(input('Enter your choice (1-3): '))
if choice == 1:
    output = 'The line of text with all its blank spaces removed:\n'
    output += "".join(string.split(" "))
elif choice == 2:
    output = 'The line of text with each word in reversed order:\n'
    stringsp = string.split(" ")
    output += stringsp[0][::-1]
    for words in stringsp[1:]:
        output += ' '
        output += words[::-1]
elif choice == 3:
    output = 'The number of words in the line of text: '
    stringsp = string.split(" ")
    count = 0
    for words in stringsp:
        if words != '':
            count += 1
    output += str(count)
else:
```

```
    output = 'Invalid input. You must type 1 or 2 or 3!'


# Output the result of the operation if the user choice of operation is
# valid; otherwise output an error message
print(output)
```

OUTPUT:

```
Enter a line of text without punctuation mark:
No snowflake    in an avalanche   ever feels responsible
Type a number 1 to 3 to do one of the following operations
 1 - print the line of text with all its blank spaces removed
 2 - print the line of text with each word in reversed order
 3 - print the number of words in the line of text
Enter your choice (1-3): 1
The line of text with all its blank spaces removed:
Nosnowflakeinanavalancheeverfeelsresponsible

Enter a line of text without punctuation mark:
No snowflake    in an avalanche   ever feels responsible
Type a number 1 to 3 to do one of the following operations
 1 - print the line of text with all its blank spaces removed
 2 - print the line of text with each word in reversed order
 3 - print the number of words in the line of text
Enter your choice (1-3): 2
The line of text with each word in reversed order:
oN ekalfwons    ni na ehcnalava   reve sleef elbisnopser

Enter a line of text without punctuation mark:
No snowflake    in an avalanche   ever feels responsible
Type a number 1 to 3 to do one of the following operations
 1 - print the line of text with all its blank spaces removed
 2 - print the line of text with each word in reversed order
 3 - print the number of words in the line of text
Enter your choice (1-3): 3
The number of words in the line of text: 8

Enter a line of text without punctuation mark:
No snowflake    in an avalanche   ever feels responsible
Type a number 1 to 3 to do one of the following operations
 1 - print the line of text with all its blank spaces removed
 2 - print the line of text with each word in reversed order
 3 - print the number of words in the line of text
Enter your choice (1-3): 0
Invalid input. You must type 1 or 2 or 3!
```

# Exercise 2: Caesar Cipher

## AIM:

A Caesar cipher is a simple substitution cipher that produces a cipher text by shifting each plain-text letter in a message a fixed number of places down the alphabet in a circular fashion so that the next character after "z" and “Z” are "a" and ”A”, respectively. The plain text is your original message while the cipher text is the encrypted message. For example, if the key value (i.e. the shift length) is 2, then the word "University Physics” would be encoded as "Wpkxgtukva Rjaukeu". The original message can be recovered by decoding the encrypted message using the reverse process.

Write a Python program that prompts for a key value in the closed interval [1, 25] and the plain text to encode from the user and then outputs the cipher text as well as the decrypted cipher text. You can assume that the input plain text consists only of letters and spaces. And your program should ask the user to input the key value again if the input value is invalid. If your program is correct, then the decrypted cipher text should match the original plain text.

(Hint: Construct two strings where one contains all the plain-text letters and one contains all the corresponding cipher-text letters. Notice that the letter to substitute is at the same index in both strings.)

## ALGORITHM:

1. Start
2. Input a key value and a plain text; ask the user to input again if the input key value is invalid
3. Construct the strings for the plain text and cipher text letters
4. Encode the plain text by substituting each letter with the corresponding cipher text letter and then output the result
5. Decode the cipher text by substituting each letter with the corresponding plain text letter and then output the result
6. End

## PROGRAM:

```
# Exercise 2: Caesar Cipher
# Written by F K Chow, HKU
# Last update: 2021/12/10

# Input a key value and a plain text. Ask the user to input again if
# the input key value is invalid
```

```
while True:
    n = int(input('Enter a key value n (1-25): '))
    if n > 0 and n < 26:
        break
    print('Invalid input! n must be an integer in the range of 1-25!')
string = str(input('Enter a plain text: '))

# Construct the strings for the plain text and cipher text letters
plaintxtlett = 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ'
ciphertxtlett = plaintxtlett[n:26]
ciphertxtlett += plaintxtlett[0:n]
ciphertxtlett += plaintxtlett[n+26:52]
ciphertxtlett += plaintxtlett[26:n+26]

# Encode the plain text and output the result
cipher_text = ''
for ch in string:
    if ch != ' ':
        index = plaintxtlett.index(ch)
        cipher_text += ciphertxtlett[index]
    else:
        cipher_text += ' '
print('The cipher text is', cipher_text)

# Decode the plain text and output the result
decryp_text=''
for ch in cipher_text:
    if ch != ' ':
        index = ciphertxtlett.index(ch)
        decryp_text += plaintxtlett[index]
    else:
        decryp_text += ' '
print('The decrypted cipher text is', decryp_text)
```

(The output is shown on the next page.)

OUTPUT:

```
Enter a key value n (1-25): 2
Enter a plain text: University Physics
The cipher text is Wpkxgtukva Rjaukeu
The decrypted cipher text is University Physics

Enter a key value n (1-25): -6
Invalid input! n must be an integer in the range of 1-25!
Enter a key value n (1-25): 18
Enter a plain text: Today is Monday
The cipher text is Lgvsq ak Egfvsq
The decrypted cipher text is Today is Monday
```

# Exercise 3: Vector Manipulation

## AIM:

Write a Python program that prompts the user for two vectors of the same length, computes their sum, difference, and dot product by manipulating the lists storing these vectors, and finally outputs the results. You can assume that each vector is input as a comma-space-separated list of integers enclosed by square brackets. And your program should ask the user to input the vectors again if they have different length. Here are the sample input and output of this program:

```
Enter the vector A: [2, 4, 6, 8]
Enter the vector B: [1, 2, 3]
Invalid input! Vectors A and B must have the same length.
Enter the vector A: [2, 4, 6, 8]
Enter the vector B: [1, 2, 3, 4]
A+B = [3, 6, 9, 12]
A-B = [1, 2, 3, 4]
A.B = 60
```

## ALGORITHM:

1. Start
2. Input two vectors and store them as lists of integers using string methods `strip` and `split`. Check whether the vectors have the same length by comparing the lengths of the lists. Ask user to input the vectors again if they have different lengths.
3. Compute the sum, difference, and dot product of these two vectors by manipulating the lists storing them via list comprehension
4. Output the results of the computation in step 3
5. End

## PROGRAM:

```
# Exercise 3: Vector Manipulation
# Written by F K Chow, HKU
# Last update: 2021/12/10

# Input two vectors of the same length and store them as lists
while True:
```

```
    strA = input('Enter the vector A: ')
    strB = input('Enter the vector B: ')
    strA = strA.strip('[]')
    strB = strB.strip('[]')
    lstA = strA.split(', ')
    lstB = strB.split(', ')
    if len(lstA) == len(lstB):
        A = [int(i) for i in lstA]
        B = [int(i) for i in lstB]
        break
    print('Invalid input! Vectors A and B must have the same length.')

# Compute the sum, difference, and dot product of these two vectors by
# manipulating the lists storing them
AplusB = [A[i]+B[i] for i in range(len(A))]
AminusB = [A[i]-B[i] for i in range(len(A))]
AdotB = sum([A[i]*B[i] for i in range(len(A))])

# Output the results
print('A+B =', AplusB)
print('A-B =', AminusB)
print('A.B =', AdotB)
```

OUTPUT:

```
Enter the vector A: [8, 7, 2, 5]
Enter the vector B: [3, 5, 8]
Invalid input! Vectors A and B must have the same length.
Enter the vector A: [8, 7, 2, 5]
Enter the vector B: [3, 5, 8, 0]
A+B = [11, 12, 10, 5]
A-B = [5, 2, -6, 5]
A.B = 75
```

# Exercise 4: Bubble Sort

## AIM:

Bubble sort is a simple algorithm that sorts a collection of data by 'bubbling' values to the end of the list over successive passes like air bubbles rising in water. Here are the procedures for sorting a list of numbers in ascending order using bubble sort:

(a) Starting from the first element, compare each adjacent pair of elements of the list in turn and swap the elements if the former element has a larger value. After all the elements in the list have been compared, the largest element will be at the end of the list. This completes the first pass of the algorithm.

(b) Repeat the process in (a) from the first to the second last element of the list. At the end of the second pass, the second largest element will be at the second last position of the list.

(c) Repeat the process in (a) again in a similar manner as in (b) until no more swapping occurs in a pass.

Write a Python program that generates a list composed of 10 random integers in the range of 1 to 100, prints out the list, and finally uses bubble sort to sort the integers in the list in ascending order with the updated list printed out after each swapping. Here is the sample output of this program:

```
Sorting Process of a List of Integers by Bubble Sort:
[83, 74, 89, 26, 18, 69, 75, 98, 47, 77]
[74, 83, 89, 26, 18, 69, 75, 98, 47, 77]
[74, 83, 26, 89, 18, 69, 75, 98, 47, 77]
[74, 83, 26, 18, 89, 69, 75, 98, 47, 77]
[74, 83, 26, 18, 69, 89, 75, 98, 47, 77]
[74, 83, 26, 18, 69, 75, 89, 98, 47, 77]
[74, 83, 26, 18, 69, 75, 89, 47, 98, 77]
[74, 83, 26, 18, 69, 75, 89, 47, 77, 98]
[74, 26, 83, 18, 69, 75, 89, 47, 77, 98]
[74, 26, 18, 83, 69, 75, 89, 47, 77, 98]
[74, 26, 18, 69, 83, 75, 89, 47, 77, 98]
[74, 26, 18, 69, 75, 83, 89, 47, 77, 98]
[74, 26, 18, 69, 75, 83, 47, 89, 77, 98]
[74, 26, 18, 69, 75, 83, 47, 77, 89, 98]
[26, 74, 18, 69, 75, 83, 47, 77, 89, 98]
[26, 18, 74, 69, 75, 83, 47, 77, 89, 98]
```

```
[26, 18, 69, 74, 75, 83, 47, 77, 89, 98]
[26, 18, 69, 74, 75, 47, 83, 77, 89, 98]
[26, 18, 69, 74, 75, 47, 77, 83, 89, 98]
[18, 26, 69, 74, 75, 47, 77, 83, 89, 98]
[18, 26, 69, 74, 47, 75, 77, 83, 89, 98]
[18, 26, 69, 47, 74, 75, 77, 83, 89, 98]
[18, 26, 47, 69, 74, 75, 77, 83, 89, 98]
```

## ALGORITHM:

1. Start
2. Import the `randint` function from the random module
3. Generate a list of 10 random integers in the range of 1 to 100 and print out the list
4. Use bubble sort to sort the integers in the list in ascending order by doing the followings for *i* = 1 to 10:
   (a) Compare each adjacent pair of elements of the list from the first element to the *i*th last element in turn. For each comparison, if the former element has a larger value, swap the two elements and then output the updated list.
   (b) If no swapping occurs in (a), then jump to step 5.
5. End

## PROGRAM:

```
# Exercise 4: Bubble Sort
# Written by F K Chow, HKU
# Last update: 2022/1/27

# Import the randint function from the random module
from random import randint

# Generate a list of 10 random integers in the range of 1 to 100 and
# print out the list
list = [randint(1, 100) for i in range(10)]
print("Sorting Process of a List of Integers by Bubble Sort:")
print(list)
```

```
# Use bubble sort to sort the integers in the list in ascending order
# with the updated list printed out after each swapping
for i in range(1, 10):
    noswapping = True
    for j in range(10-i):
        if list[j] > list[j+1]:
            temp = list[j]
            list[j] = list[j+1]
            list[j+1] = temp
            print(list)
            noswapping = False
    if noswapping:
        break
```

OUTPUT:

```
Sorting Process of a List of Integers by Bubble Sort:
[18, 89, 43, 31, 46, 3, 9, 45, 6, 55]
[18, 43, 89, 31, 46, 3, 9, 45, 6, 55]
[18, 43, 31, 89, 46, 3, 9, 45, 6, 55]
[18, 43, 31, 46, 89, 3, 9, 45, 6, 55]
[18, 43, 31, 46, 3, 89, 9, 45, 6, 55]
[18, 43, 31, 46, 3, 9, 89, 45, 6, 55]
[18, 43, 31, 46, 3, 9, 45, 89, 6, 55]
[18, 43, 31, 46, 3, 9, 45, 6, 89, 55]
[18, 43, 31, 46, 3, 9, 45, 6, 55, 89]
[18, 31, 43, 46, 3, 9, 45, 6, 55, 89]
[18, 31, 43, 3, 46, 9, 45, 6, 55, 89]
[18, 31, 43, 3, 9, 46, 45, 6, 55, 89]
[18, 31, 43, 3, 9, 45, 46, 6, 55, 89]
[18, 31, 43, 3, 9, 45, 6, 46, 55, 89]
[18, 31, 3, 43, 9, 45, 6, 46, 55, 89]
[18, 31, 3, 9, 43, 45, 6, 46, 55, 89]
[18, 31, 3, 9, 43, 6, 45, 46, 55, 89]
[18, 3, 31, 9, 43, 6, 45, 46, 55, 89]
[18, 3, 9, 31, 43, 6, 45, 46, 55, 89]
[18, 3, 9, 31, 6, 43, 45, 46, 55, 89]
[3, 18, 9, 31, 6, 43, 45, 46, 55, 89]
[3, 9, 18, 31, 6, 43, 45, 46, 55, 89]
[3, 9, 18, 6, 31, 43, 45, 46, 55, 89]
[3, 9, 6, 18, 31, 43, 45, 46, 55, 89]
[3, 6, 9, 18, 31, 43, 45, 46, 55, 89]
```

# Exercise 5: Printing a Histogram

## AIM:

Below is the table of the accommodation in government primary schools by district in Hong Kong, 2019 (Source: Student Enrolment Statistics, 2019/20 by Education Bureau).

| District | Numbers of Students (nearest hundreds) | District | Numbers of Students (nearest hundreds) |
|---|---|---|---|
| Central & Western | 7 | Sai Kung | 7 |
| Wan Chai | 25 | Sha Tin | 8 |
| Eastern | 22 | Tai Po | 8 |
| Southern | 8 | North | 8 |
| Yau Tsim Mong | 14 | Yuen Long | 27 |
| Sham Shui Po | 22 | Tuen Mun | 7 |
| Kowloon City | 29 | Tsuen Wan | 18 |
| Wong Tai Sin | 6 | Kwai Tsing | 0 |
| Kwun Tong | 15 | Islands | 0 |

Write a Python program to show the statistics of the data in this table by printing a histogram in the following format:

```
Accommodation in Government Primary Schools by Districts in HK, 2019
(in nearest hundreds)
Central & Western | ******* (7)
          Wan Chai | ************************* (25)
                   ⋮
           Islands | (0)
```

Your program should define some variables to store the given data and then print the variables using `print` with the format specifier.

## ALGORITHM:

1. Start
2. Define two lists to store the district names and the number of students in each district
3. Print a histogram to show the statistics of the given data by using `print` with the format specifier

4. End

PROGRAM:

```
# Exercise 5: Printing a Histogram
# Written by F K Chow, HKU
# Last update: 2022/1/27

# Define two lists to store the district names and the number of
# students in each district
district = ['Central & Western', 'Wan Chai', 'Eastern', 'Southern',
            'Yau Tsim Mong', 'Sham Shui Po', ' Kowloon City',
            'Wong Tai Sin', 'Kwun Tong', 'Sai Kung', 'Sha Tin',
            'Tai Po', ' North', 'Yuen Long', 'Tuen Mun', 'Tsuen Wan',
            'Kwai Tsing', 'Island']
studentnum = [7, 25, 22, 8, 14, 22, 29, 6, 15, 7, 8, 8, 8, 27, 7, 18,
              0, 0]

# Print a histogram to show the statistics of the given data
print('Accommodation in Government Primary Schools by Districts in \
HK, 2019')
print('(in nearest hundreds)')
for i in range(len(district)):
    print('{:>17s} | '.format(district[i]), end='')
    if studentnum[i] != 0:
        print('{:s} ({:d})'.format('*'*studentnum[i], studentnum[i]))
    else:
        print('(0)')
```

(The output is shown on the next page.)

OUTPUT:

```
Accommodation in Government Primary Schools by Districts in HK, 2019
(in nearest hundreds)
Central & Western | ******* (7)
         Wan Chai | ************************* (25)
          Eastern | ********************** (22)
         Southern | ******** (8)
    Yau Tsim Mong | ************** (14)
     Sham Shui Po | ********************** (22)
     Kowloon City | ****************************** (29)
     Wong Tai Sin | ****** (6)
        Kwun Tong | *************** (15)
         Sai Kung | ******* (7)
          Sha Tin | ******** (8)
           Tai Po | ******** (8)
            North | ******** (8)
        Yuen Long | ************************** (27)
         Tuen Mun | ******* (7)
        Tsuen Wan | ****************** (18)
       Kwai Tsing | (0)
           Island | (0)
```