INSTITUTO POLITECNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO (ESCOM)

TEORIA COMPUTACIONAL

NOMBRE Y NÚMERO DE LA PRÁCTICA:

- PRÁCTICA 2. MANEJO DE CADENAS EN PYTHON

NOMBRE DEL ALUMNO:

- SANTOS MÉNDEZ ULISES JESÚS

NOMBRE DEL MAESTRO:

- JORGE LUIS ROSAS TRIGUEROS

FECHA DE REALIZACIÓN:

- 23/10/2020

FECHA DE ENTREGA:

- 6/11/2020

Marco teórico:

El tipo de datos cadena

Una cadena es una secuencia de caracteres (letras, números, espacios, marcas de puntuación, etc.) y en Python se distingue porque va encerrada entre comillas simples o dobles.

Las cadenas pueden usarse para representar información textual: nombres de personas, nombres de colores, matrículas de coche, etc.

Las cadenas también pueden almacenarse en variables (Véase figura 1)[1] por ejemplo:

```
>>> nombre = 'Pepe' ↲
>>> nombre ↲
'Pepe'
```

(Figura 1)

Es posible realizar operaciones con cadenas (Véase figura 2)[2] como:

- Suma
- Resta

```
>>> 'a' + 'b' ↲
'ab'
>>> nombre = 'Pepe' ↲
>>> nombre + 'Cano' ↲
'PepeCano'
>>> nombre + '␣' + 'Cano' ↲
'Pepe Cano'
>>> apellido = 'Cano' ↲
>>> nombre + '␣' + apellido ↲
'Pepe Cano'
```

(Figura 2)

Métodos

Los datos de ciertos tipos permiten invocar unas funciones especiales: los denominados "métodos". De los que solo cadenas permiten invocar métodos sobre ellas.

La sintaxis es diferente de la propia de una llamada a función convencional. Lo primero que aparece es el propio objeto sobre el que se efectúa la llamada. El nombre del método se separa del objeto con un punto. Los paréntesis abiertos y cerrados son obligatorios (Véase figura 3)[3].

Los métodos de las cadenas fueron funciones de módulos en versiones antiguas de Python, algunos métodos aceptan parámetros, el método busca el patrón en la cadena sobre la que se invoca el método.

```
>>> 'un␣pequeño␣ejemplo'.replace('pequeño', 'gran') ↲
'un gran ejemplo'
>>> una_cadena = 'abc'.replace('b', '-') ↲
>>> una_cadena ↲
'a-c'
```

(Figura 3)

[1] Figura 1: se muestra como se guarda una cadena en una variable en el IDE con Python.
[2] Figura 2: S e muestra cómo se hace una suma con cadenas y a su vez se guardan en variables.
[3] Figura 3: Se muestra como a la cadena se le asigna el objeto .replace, este método busca remplazar algún carácter de la cadena

Material y Equipo:

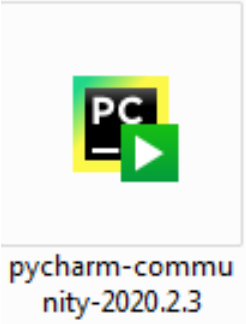
-PC (véase figura 4)[4]

(Figura 4)

-Python 3 (Python 3.8.6) (véase figura. 5)[5]

(Figura 5)

-IDE (Pycharm) (véase figura 6)[6]



(Figura 6)

[4] Figura 4: PC o cualquier computadora para el desarrollo de la práctica
[5] Figura 5:Ejecutable para instalar Python 3.8.6
[6] Figura 6:Ejecutable de IDE Pycharm

1) Se dejó una asignación en Microsoft Teams sobre la práctica y consiste en el manejo de cadenas en Python, se sugirió hacer los programas en un solo archivo, utilice el manejo de funciones vacías y con paso de parámetros.
   Cada programa le corresponde una función.
   Las problemáticas presentadas fue el cómo manejar funciones en Python, y el pensar si sería posible poner todos los programas en 1 solo archivo plano, al final la práctica se llegó a la idea que es posible optimizar el programa en general al meter un switch.

Prac_2.py

```python
import re
from collections import Counter
cadena = input("Ingrese su cadena: ")
patron = "0123456789"
fuera = 'aeiouAEIOU'
alfabeto = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
lista = ["Lunes", "Martes", "Miercoles", "Jueves", "Viernes", "Sabado", "Domingo"]
# Programa 1


def longitud():
    print("La longitud de tu cadena es: ", len(cadena))


longitud()


# Programa 2
def numcar():
    print(Counter(cadena))


numcar()


# Programa 3
def newcad():
    nueva = cadena[:2]+cadena[-2:]
    if len(cadena) < 2:
        print("{}")
    else:
        print(nueva)
```

Prac_2.py

```python

# Programa 4
def cambcar():
    cambio = cadena[0]+cadena[1:].replace(cadena[0], '$')
    print(cambio)


cambcar()


# Programa 5
def camcad():
    ocadena = input("Ingresa otra cadena: ")
    cambio1 = cadena.replace(cadena[:2], ocadena[:2])
    cambio2 = ocadena.replace(ocadena[:2], cadena[:2])
    print(cambio1, " ", cambio2)


camcad()


# Programa 6
def mostlarge(lis):
    print(lis)
    mas_larga = max([len(c) for c in lis])
    return mas_larga


print("La longitud mas larga de la lista es: ", mostlarge(lista))
```

Prac_2.py

```
67
68  # Programa 7
69  def quitarcar():
70      quitar = int(input("Ingrese la posicion que desee quitar: "))
71      elim = cadena.replace(cadena[quitar], ' ')
72      print(elim)
73
74
75  quitarcar()
76
77
78  # Programa 8
79  def cambio_car(text):
80      return text[-1] + text[1:-1] + text[0]
81
82
83  print(cambio_car(cadena))
84
85
86  # Programa 9
87  def impar(cad):
88      txt = ''
89      for i in range(len(cad)):
90          if i % 2 == 0:
91              txt += cad[i]
92      return txt
93
94
95  print(impar(cadena))
```

Prac_2.py

```
97
98  # Programa 10
99  def word():
100     palabra = cadena.split(' ')
101     frecuencia = [palabra.count(p) for p in palabra]
102     print(str(list(zip(palabra, frecuencia))))
103
104
105 word()
106
107
108 # Programa 11
109 def maymin():
110     mayus = cadena.upper()
111     minus = cadena.lower()
112     print("Tu cadena en mayusculas es: ", mayus)
113     print("Tu cadena en minusculas es: ", minus)
114
115
116 maymin()
117
118
119 # Programa 12
120 def alfnum():
121     entrada = input("Ingresa palabras separadas por ,:")
122     paso1 = entrada.split(',')
123     print(paso1)
124
125
126 alfnum()
```

```python
# Programa 13
def fourcopies():
    tam = len(cadena)
    last = cadena[tam-2:]
    for c in range(4):
        print(last, end='')


fourcopies()


# Programa 14
def firsttres():
    tam = len(cadena)
    if tam <= 3:
        print("\n", cadena)
    else:
        print("\n", cadena[:3])


firsttres()
```

```python
# Programa 15
def mitad():
    tam = len(cadena)
    if tam % 2 == 0:
        print(cadena[:tam//2])
    else:
        print("La cadena es impar")
        print(cadena)


mitad()


# Programa 16
def inversa():
    print("Mi numero de boleta es : 2020630460")
    tama = len(cadena)
    sumita = 4 + 6 + 0
    if tama == tama * sumita:
        print(cadena[::-1])
    else:
        print("No es multiplo de la suma de los ultimos tres digitos de la boleta ")
        print(cadena)


inversa()
```

```python
# Programa 17
def mayusc():
    if len(cadena) > 4:
        conta = 0
        for i in range(4):
            if cadena[i] == cadena[i].upper():
                conta += 1
        if conta >= 2:
            print(cadena.upper())
        else:
            print(cadena)
    else:
        return cadena


mayusc()


# Programa 18
def lexico():
    print(''.join(sorted(cadena, key=str.upper)))


lexico()
```

```python
# Programa 19
def especifico():
    status = 0
    print("La cadena es: ", cadena)
    while status != 1:
        print("¿Con que caracter comienza la cadena?")
        caracter = input()
        if caracter == cadena[0]:
            print("Afirmativo, la cadena comienza con \'", caracter, "\'")
            status = 1
            print("Programa finalizado. . .")
        else:
            print("La cadena no empieza con \'", caracter, "\'\nVuelva a intentar.")
            status = 0
            print("volviendo al principio. . .\n\n")


especifico()
```

```python
# Programa 21
def multil():
    cadena_multilinea = '''uno
    dos
    tres
    cuatro
    '''
    print('La cadena multilinea original es:\n', cadena_multilinea)
    prefijo = 'HOLA'
    print('El prefijo es: ', prefijo)
    renglones = cadena_multilinea.split()
    print()
    for n in range(len(renglones)):
        renglones[n] = prefijo + renglones[n]
    print('La cadena multilinea con el prefijo al principio de cada linea es: ')
    cadena_multilinea_con_prefijo = ('\n'.join(renglones))
    print(cadena_multilinea_con_prefijo)


multil()
```

```
# Programa 22
def flotantes():
    decimal = 87.329765
    print("El flotante es: ", decimal, "\nEl flotante limitado es: %.2f" % decimal)


flotantes()


# Programa 23
def sign():
    decimal = 2.718281
    decimal2 = - 5581.199513
    print("Tu flotante es: ", decimal)
    print('Tu flotante con signo es: {:+.2f}'.format(decimal))
    print("Tu flotante es: ", decimal2)
    print('Tu flotante con signo es: {:+.2f}'.format(decimal2))


sign()


# Programa 24
def nodec():
    decimal = 7.382985
    print("Tu flotante es: ", decimal)
    print("Tu flotante limitado es %.0f" % decimal)


nodec()
```

```python
# Programa 25
def cerosnum():
    ceros = ' '
    nums = '302939'
    ancho = int(input("Ingresa el ancho de ceros a agregar: "))
    for i in range(ancho):
        ceros += '0'
    final = ceros + nums
    print(final)


cerosnum()


# Programa 26
def estrellas():
    var = ' '
    num = input("Ingresa algunos numeros: ")
    ancho = int(input("Ingresa un ancho especificado: "))
    for c in range(ancho):
        var += '*'
    print(num + var)


estrellas()
```

```python
# Programa 27
def comas():
    num = int(input("Ingrese un numero: "))
    print('{:,}'.format(num))


comas()


# Programa 28
def porcentaje():
    num = float(input("Ingrese un numero: "))
    if num < 1:
        numper = 'El porcentaje de {} es: {:.2%}'.format(num, num)
        print(numper)
    else:
        print(num, "%")


porcentaje()
```

```python
# Programa 29
def align():
    num = input("Ingresa un numero: ")
    print("Original: ", num)
    print("Izq:", format(num, '<10'))
    print("Der:", format(num, '>10'))
    print("Cen:", format(num, '^10'))


align()


# Programa 30
def rgb():
    rojo = int(input("Ingrese el color rojo (0,256): "))
    verde = int(input("Ingrese el color verde (0,256): "))
    azul = int(input("Ingrese el color azul (0,256): "))
    if (0 <= rojo < 256) and (0 <= verde < 256) and (0 <= azul < 256):
        rgb1 = [rojo, verde, azul]
        rgb_cad = f'#{rgb1[0]:02x}{rgb1[1]:02x}{rgb1[2]:02x}'
        print("Tu valor hexadecimal es: ", rgb_cad)
    else:
        print("No se encuentra en el rango pedido")


rgb()
```

```python
# Programa 31
def ocurr():
    subcadena = "ata"
    ocurrencias = cadena.count(subcadena)
    print("Cadena original: \'", cadena, "\'")
    print("Subcadena a buscar: \'", subcadena, "\'")
    print("El número de ocurrencias de la subcadena en la cadena es: ", ocurrencias)


ocurr()


# Programa 32
def invcad():
    print(cadena[::-1])


invcad()


# Programa 33
def supr():
    nueva = ''.join([c for c in cadena if c not in fuera])
    print(nueva)


supr()
```

```python
# Programa 34
def repetidos():
    contar = Counter(cadena)
    duplicados = [t[1] for t in list(contar.items()) if t[1] > 1]

    print(duplicados)


repetidos()


# Programa 35
def caractual():
    n = 0
    while n != len(cadena):
        print("Carácter actual: ", cadena[n], " posición ", n)
        n = n + 1


caractual()
```

```python
# Programa 36
def coinalfab():
    valor = 0
    for p in alfabeto:
        if re.search(p, cadena):
            valor = 1
        else:
            valor = 0
    if valor == 1:
        print("Contiene todas las letras del alfabeto")
    else:
        print("No contiene todas las letras del alfabeto")


coinalfab()


# Programa 37
def cadtolis():
    print("Cadena original: ", cadena)
    renglones = cadena.split()
    print("\nImpresión de la lista:", renglones)


cadtolis()
```

```python
# Programa 38
def minusculas():
    n = int(input("Ingresa a n: "))
    res = cadena[:n].lower() + cadena[n:]
    print("Cadena original: ", cadena)
    print("Cadena con n minúsculas: ", res)


minusculas()


# Programa 39
def campyc():
    punto = ".,"
    coma = ",."
    cambio = cadena.maketrans(punto, coma)
    print(cadena.translate(cambio))


campyc()
```

```python
# Programa 40
def convocales():
    res = set([c for c in cadena if c in fuera])
    print(res)
    print(len(res))


convocales()


# Programa 41
def ultimate():
    regreso = cadena.rsplit('n', maxsplit=1)
    print(regreso)


ultimate()


# Programa 42
def antescar():
    print(cadena.rsplit("e", 1))


antescar()
```

```python
# Programa 43
def fijos():
    for n in range(len(cadena) + 1):
        print("\nSubcadena ", n + 1, "=", cadena[:n], end=" ")
        if cadena[:n] != cadena:
            print("Prefijo propio")
        else:
            print("Prefijo")
    for i in range(len(cadena), -1, -1):
        print("\nSubcadena ", -i + (len(cadena) + 1), "=", cadena[i:], end=" ")
        if cadena[i:] != cadena:
            print("Sufijo propio")
        else:
            print("Sufijo")


fijos()
```

2) Ejecución del programa en general con todas las salidas en consola, en la gran mayoría se utilizó como variable global a “cadena” y posteriormente en las funciones se utilizó una variable local para almacenar el ingreso de un número o una cadena, en su mayoría fueron cadenas.

```
"F:\Teoria Computacional\Programas\P2\Scripts\python.exe" "F:/Teoria Computacional/Programas/P2/Prac_2.py"
Ingrese su cadena: ULISES
La longitud de tu cadena es:  6
Counter({'S': 2, 'U': 1, 'L': 1, 'I': 1, 'E': 1})
ULES
ULISES
Ingresa otra cadena: santos
saISES   ULntos
['Lunes', 'Martes', 'Miercoles', 'Jueves', 'Viernes', 'Sabado', 'Domingo']
La longitud mas larga de la lista es:  9
Ingrese la posicion que desee quitar: 3
ULI E
SLISEU
UIE
[('ULISES', 1)]
Tu cadena en mayusculas es:  ULISES
Tu cadena en minusculas es:  ulises
Ingresa palabras separadas por ,:escuela,superior,computo
['escuela', 'superior', 'computo']
ESESESES
 ULI
ULI
Mi numero de boleta es : 2020630460
No es multiplo de la suma de los ultimos tres digitos de la boleta
ULISES
ULISES
EILSSU
La cadena es:  ULISES
¿Con que caracter comienza la cadena?
U
Afirmativo, la cadena comienza con ' U '
Programa finalizado. . .
La cadena multilinea original es:
 uno
    dos
    tres
    cuatro

El prefijo es:  HOLA
```

```
La cadena multilinea con el prefijo al principio de cada linea es:
HOLAuno
HOLAdos
HOLAtres
HOLAcuatro
El flotante es:  87.329765
El flotante limitado es: 87.33
Tu flotante es:  2.718281
Tu flotante con signo es: +2.72
Tu flotante es:  -5581.199513
Tu flotante con signo es: -5581.20
Tu flotante es:  7.382985
Tu flotante limitado es 7
Ingresa el ancho de ceros a agregar: 6
 000000302939
Ingresa algunos numeros: 54678654
Ingresa un ancho especificado: 9
54678654 *********
Ingrese un numero: 5581199513
5,581,199,513
Ingrese un numero: 0.36
El porcentaje de 0.36 es: 36.00%
Ingresa un numero: 65
Original:  65
Izq: 65
Der:         65
Cen:     65
Ingrese el color rojo (0,256): 120
Ingrese el color verde (0,256): 33
Ingrese el color azul (0,256): 44
Tu valor hexadecimal es:  #78212c
Cadena original: ' ULISES '
Subcadena a buscar: ' ata '
El número de ocurrencias de la subcadena en la cadena es:  0
SESILU
LSS
[2]
```

```
Carácter actual:  U  posición  0
Carácter actual:  L  posición  1
Carácter actual:  I  posición  2
Carácter actual:  S  posición  3
Carácter actual:  E  posición  4
Carácter actual:  S  posición  5
No contiene todas las letras del alfabeto
Cadena original:  ULISES

Impresión de la lista: ['ULISES']
Ingresa a n: 3
Cadena original:  ULISES
Cadena con n minúsculas:  uliSES
ULISES
{'I', 'E', 'U'}
3
['ULISES']
['ULISES']
```

```
Subcadena  1 =  Prefijo propio

Subcadena  2 = U Prefijo propio

Subcadena  3 = UL Prefijo propio

Subcadena  4 = ULI Prefijo propio

Subcadena  5 = ULIS Prefijo propio

Subcadena  6 = ULISE Prefijo propio

Subcadena  7 = ULISES Prefijo

Subcadena  1 =  Sufijo propio

Subcadena  2 = S Sufijo propio

Subcadena  3 = ES Sufijo propio

Subcadena  4 = SES Sufijo propio

Subcadena  5 = ISES Sufijo propio

Subcadena  6 = LISES Sufijo propio

Subcadena  7 = ULISES Sufijo

Process finished with exit code 0
```

Conclusiones:

En conclusión esta práctica fue muy importante para conocer los distintos comandos y herramientas que nos brinda Python para el manejo de cadenas así como darle formato a una cadena o a un texto, también se pueden hacer conversiones utilizando librerías que se importan, fue necesaria la elaboración de la práctica para la familiarización con Python y el IDE que se está utilizando.

Bibliografía:

- *Practique la teoría de autómatas y lenguajes formales*, Leonardo Alonso Hernández Rodríguez, 2010.
- *Python para todos*, Raúl González Duque, 2007.
- *Introducción a la programación en Python*, Andrés Marzal, Isabel García, 2010.