PONTIFICIA
UNIVERSIDAD
CATÓLICA
DE CHILE

INTRODUCCIÓN A LA PROGRAMACIÓN

MINICONTROL
6 de agosto

SOLEMNE 1 (18%)
19 de agosto

CONTROL 2 (12%)
3 de septiembre

Luego del 18, solemne 2
y examen

Strings

- Objetivo:
  - Introducir tema: strings (cadenas de caracteres)
  - Cubrir tema completo
- Temario
  - Recuento *control de flujo*
    - Instrucciones If-elif-else, for-range, while
  - Capacidades de strings
  - **Índices**, **slices**, **for** para strings
  - Substrings

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Control de flujo

Alternativas. Ejemplo 1:

```
x = input()
if x == "auto":
    print("Escribiste",x)
```

Alternativas. Ejemplo 2:

```
x = int(input())
if x > 0:
    print("Nro positivo")
elif x < 0:
    print("Nro negativo")
else:
    print("Cero")
```

While. Ejemplo 1:

```
i = 10
while i > 0:
   print("T -",i)
   i = i-1
print("Despegue!")
```

While. Ejemplo 2:

```
x = "si"
while x == "si":
  print("En el while.")
  print("Continuar?(si)")
  x = input()
  print()
print("Salimos.")
```

While. Ejemplo 3:

```
suma = 0
seguir = True
while seguir:
  x = input()
  if x == "":
    seguir = False
  else:
    suma = suma +
float(x)
print("Suma:", suma)
```

For-range. Ejemplo 1:

```
for i in range(15):
  print("Iteracion",i)
```

For-range. Ejemplo 2:

```
for i in range(10):
  print("T -", 10-i)
print("Despegue!")
```

For-range. Ejemplo 3:

```
print("Escriba un nro")
nro = int(input())
dvs = 0
for i in range(2,nro):
  if nro%i == 0:
    print(i,"es divisor")
    dvs = dvs + 1
print("el numero", nro,
    "tiene",dvs,divisores)
```

Elemental de string

x + y – concatena los strings **x** e **y**

x in y – True si el string **x** está contenido dentro de **y**

len(x) – retorna el largo de **x**

x[n] – retorna el caracter en el índice **n** dentro de **x**; **n < len(x)**. Si **n < 0**, se cuenta desde el final

Slices de strings

x[i:j] – retorna una porción del string **x**, desde el índice **i** al índice **j**, sin tocarlo (como si fuera **range**)

x[:j] – del inicio al índice **j-1**

x[i:] – del índice i al final

Recorrer strings (for)

```
txt = input()
# caracteres en orden
for c in txt:
        print(c)
# ahora pero con indices
for i in range( len(txt) ):
        print(i, txt[i])
# de fin a inicio
for i in  range(1,1+len(txt)):
        print(-i, txt[-i])
```

Localiza substrings

x.index(y) – retorna primer índice en el cual **y** parte dentro de **x**; sino, lanza error

x.find(y) – retorna primer índice en el cual **y** parte dentro de **x**; si no está, -1

x.count(y) – retorna el número de veces que **y** aparece dentro de **x**

~~**x.contains(y)** – retorna **True** si **y** está contenida en **x** (lo mismo que **y in x**)~~

Transforma strings

x.strip() – elimina espacios al inicio y al final de **x**

x.replace(u,v) – retorna una copia de **x**, pero reemplazando cada ocurrencia de **u** por **v**

x.upper() – retorna una copia en mayúsculas del string **x**

x.lower() – retorna una copia en minúsculas del string **x**

x.title() – retorna una copia de **x** con cada palabra comenzando en mayúsculas y seguida de minúsculas

Verificar forma de string

x.isupper() – retorna **True** si **x** está en mayúsculas

x.islower() – retorna **True** si **x** está en minúsculas

x.isdigit() – retorna **True** si **x** sólo tiene dígitos

x.isnumeric() – funciona igual que **x.isdigit()**

x.isalpha() – retorna **True** si **x** sólo tiene letras

x.isalnum() – retorna **True** si **x** sólo tiene dígitos y/o letras

Caracteres de escape

\n (salto de línea)

\' **\"** (comillas)

\t (salto de tabulación)

**** (backslash)

CAPACIDADES DE STRING

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Acceder a caracteres según su índice

h	o	l	a		m	u	n	d	o
0	1	2	3	4	5	6	7	8	9
-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

```
x = "hola mundo"
```

- ¿Cuál es el largo del string x?
  - **`len(x)`**
    (entrega 10)

- ¿Qué carácter (símbolo) está en la posición 3?
  - **`x[3]`** (entrega "a")

- ¿Qué carácter está al final de x?
  - **`x[-1]`** (entrega "o")

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Extrayendo substrings usando índices

h	o	l	a		m	u	n	d	o
0	1	2	3	4	5	6	7	8	9
-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

- **Substring**: porción de un string
- Formato básico para extraer un substring desde el índice **i** hasta el índice **j-1**:

  x[i : j]

```
> x = "hola mundo"

> x[0:4]
"hola"

> x[ :4]
"hola"

> x[:-6]
"hola"

> x[ : ]
"hola mundo

> x[-8:-3]
"la mu"
```

```
> x[5:10]
"mundo"

> x[5: ]
"mundo"

> x[-5:]
"mundo"

> x[0:0]
""

> x[-5:8]
"mun"
```

Relación con for-range

Recorrer strings con for
Los strings son secuencias de caracteres. La instrucción for puede acceder a cada caracter del string, desde la izquierda y hacia la derecha.

```
for x in "Textual":
  print(x)
```

```
T
e
x
t
u
a
l
```

Recorrer strings con for-range
La instrucción range genera naturalmente índices válidos para recorrer strings; notemos los índices 0, 1, ..., n-1 cubren todo un string de largo n.

```
T = "Textual"

for j in range(len(T)):
  print(T[j], "en", j)
```

```
T en 0
e en 1
x en 2
t en 3
u en 4
a en 5
l en 6
```

Relación con for-range

Slices versus for-range
Es posible reproducir un slice con for-range, tal como se ve en este ejemplo.

```
T = "ANTIPATIA"
print( T[2:6] )

u = ""
for k in range(2,6):
    u = u + T[k]
print( u )
```

```
TIPA
TIPA
```

Slices con índices inválidos
Tal como con **range**, si los índices de un **slice** no llevan a seleccionar caracteres, no hay error; el resultado será simplemente un string vacío.

```
T = "Kame-Hame-HAAA"
print('1)', T[0:4] )
print('2)', T[10:14] )
print('3)', T[5:4] )
print('4)', T[5:5] )
print('5)', T[5:6] )
print('6)', T[5:7] )
print('7)', T[500:508])
```

```
1) Kame
2) HAAA
3)
4)
5) H
6) Ha
7)
```

Podemos detectar substrings (in)

- *Substring:* texto que está contenido en otro texto
  - Ej. "hola" es parte de "hola mundo"
- La instrucción es **in**, que detecta pertenencia
  - **in** entrega True/False

```
> "pikachu" in "evil pikachu"
True

> "good" in "evil pikachu"
False
```

- La detección es sensible a mayúsculas, minúsculas, etc.
- El calce debe ser **exacto**

```
> "piKachU" in "evil pikachu"
False

> "pikachu " in "evil pikachu"
False

> "pika chu" in "evil pikachu"
False
```

Usando in dentro de if, while, etc

- Ejemplo:

```
a = input("texto1: ")
b = input("texto2: ")

if a in b:
  print(a, "está contenido en", b)
elif b in a:
  print(b, "está contenido en", a)
else:
  print("Los textos son excluyentes")
```

- Ejemplos de ejecución:

```
texto1: gato
texto2: arigato
gato está contenido en arigato

texto1: arigato
texto2: gato
gato está contenido en arigato

texto1: gato
texto2: perro
Los textos son excluyentes
```

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Detectar substrings

Redunda: contains es in
El método contains de string hace lo mismo que in. Pero in es más sencillo.

```
txt = input()

if '@' in txt:
    print('hay arroba')

if txt.contains('$'):
    print('hay peso')
```

```
Pe$o Arr@ba
hay arroba
hay peso
```

Contar ocurrencias
El método **count** cuenta la frecuencia de un substring en el string. Si arroja cero, es que el substring no es tal.

```
txt = 'Batalla'

x = txt.count('a')
y = txt.count('b')
z = txt.count('ata')
print('Hay', x, 'a-es')
print('Hay', y, 'b-es')
print('Hay', z,'ata-s')
```

```
Hay 3 a-es
Hay 0 b-es
Hay 1 ata-s
```

Encontrar substrings

Podemos encontrarlos

Con el método **index** podemos encontrar substrings. Si el substring está, obtenemos su primer índice. Si no, se lanza un **error**.

```
txt = 'pan jamon queso'

i = txt.index('jam')
j = txt.index('eso')

print('jam empieza en',i)
print('eso empieza en',j)
```

```
jam empieza en 4
eso empieza en 12
```

Podemos reemplazarlos

El método **replace** permite reemplazar toda aparición de un substring por otro. El resultado es un nuevo string.

```
tt = 'pan jamon queso'
print(tt)
x = tt.replace('a','4')
print(x)
y = x.replace('e','E')
print(y)
z = y.replace('o','@')
print(z)
```

```
pan jamon queso
p4n j4mon queso
p4n j4mon quEso
p4n j4m@n quEs@
```

Problemas

- Escriba un programa cuyo input sea un string y que imprima en pantalla cuántos espacios tiene
- Escriba un programa cuyo input sea un string y que imprima en pantalla cuántas vocales y cuántos dígitos tiene (en líneas separadas)
- Escriba un programa que reciba dos strings y que imprima en pantalla si los strings son iguales, si primero está contenido en el segundo y viceversa

Problemas

- Escriba un programa que reciba un string y que imprima ese string pero cuyas vocales han sido reemplazas por números
  - La **a** se reemplaza por **4**
  - La **e** se reemplaza por **3**
  - La **i** se reemplaza por **1**
  - La **o** se reemplaza por **0**
- Escriba un programa que reciba un string y que imprima su forma reversa
  - Ante el input **dracula**, debe imprimir **alucard**
  - Ante el input **cadete**, debe imprimir **etedac**
  - Ante el input **fondas**, debe imprimir **sadnof**