



**22A0111053 : Pemrograman**

---

Pertemuan 4-5 : Rangkaian Karakter (string)

**Zaitun, S.Si., M.Mat.**

# RANGKAIAN TEKS

- Variabel untuk teks (string)
- String literal
- f-String
- Escape character
- String sebagai array
- Penggabungan dan perulangan pada string
- Mengindeks dan mengiris batas
- Operator string
- Metode string
- Format string Lanjutan



# VARIABEL UNTUK TEKS (STRING)

- String adalah urutan **karakter** yang diperlukan sebagai satu item.
- Semua karakter yang ada dalam keyboard dapat dibuat dalam bentuk tipe data teks (string)

```
print('pengantar pemrograman')
```

```
>> pengantar pemrograman
```



## KOMENTAR

Dalam python, terdapat tulisan yang tidak akan dieksekusi. Ini disebut komentar.

```
# kode ini tidak dieksekusi, dalam satu baris  
print('Hallo, Ini Saya')
```

```
"""
```

Ini komentar dalam baris banyak, tidak akan dieksekusi

```
"""
```

```
>> Hallo, Ini Saya
```



# F-STRING

- Melakukan format string dengan cara ringkas.
- Dawali dengan "f" sebelum string

```
prodi = 'Sistem Informasi'  
angkatan = 2023  
print(f'Prodi {prodi} Angkatan {angkatan}')
```

```
>> Prodi Sistem Informasi Angkatan 2023
```



# STRING LITERAL

Penulisan karakter teks (string) harus ditulis dalam literal atau dikelilingi oleh karakter kutip tunggal ('') atau kutip ganda ("") yang konsisten.

```
print('pengantar pemrograman')
print("pengantar pemrograman")
```

```
>> pengantar pemrograman
    pengantar pemrograman
```



# ESCAPE CHARACTER

Dalam penulisan teks (string) terdapat karakter yang ilegal yang perlu dipanggil khusus. Pemanggilan khusus ini disebut **escape**, biasanya dituliskan sebagai garis miring terbalik (\) diikuti oleh karakter.

```
print('Jum' at')
```

```
>>     print('Juma'at')
                  ^
SyntaxError: invalid syntax
```



# LANJUTAN 1

```
print('Jum\'at')
```

```
>> Jum'at
```

```
print('Menggunakan \"Python\"')
print("Menggunakan \"Python\"")
```

```
>> Menggunakan "Python"
>> Menggunakan "Python"
```



# LANJUTAN 2

Escape Jenis Karakter	
Code	Keterangan
\'	Single Quote
\\	Backslash
\n	New Line
\r	Carriage Return
\t	Tab
\b	Backspace
\ooo	Octal Value
\xhh	Hex Value



# LANJUTAN 3

- Single Quote

```
print('Jum\'at')
```

```
>> Jum'at
```

- Backslash

```
print('Mencoba masukkan backslash \\')
```

```
>> Mencoba masukkan backslash \
```



# LANJUTAN 4

- New Line

```
print('Pengantar\nPemrograman')
```

```
>> Pengantar  
Pemrograman
```

- Carriage Return

```
print('Pengantar\rPemrograman')
```

```
>> Pemrograman
```



# LANJUTAN 5

- Tab

```
print('Pengantar\tPemrograman')
```

```
>> Pengantar      Pemrograman
```

- Backspace

```
print('Pengantar\bPemrograman')
```

```
>> PengantaPemrograman
```



# LANJUTAN 6

- Octal Value

```
print('\104\165\156\151\141')
```

```
>> Dunia
```

- Hex Value

```
print('\x44\x75\x6e\x69\x61')
```

```
>> Dunia
```



<b>Binary</b>	<b>Oct</b>	<b>Dec</b>	<b>Hex</b>	<b>Glyph</b>	<b>Binary</b>	<b>Oct</b>	<b>Dec</b>	<b>Hex</b>	<b>Glyph</b>	<b>Binary</b>	<b>Oct</b>	<b>Dec</b>	<b>Hex</b>	<b>Glyph</b>
010 0C00	040	32	20	sr	100 0000	100	64	40	@	110 0C00	140	93	60	`
010 0C01	041	33	21	!	100 0001	101	65	41	A	110 0C01	141	97	61	€
010 0U1U	042	34	22	"	100 0U1U	102	66	42	B	110 0U1U	142	93	62	₭
010 0C11	043	35	23	#	100 0011	103	67	43	C	110 0C11	143	93	63	₵
010 0100	044	33	24	\$	100 0100	104	68	44	D	110 0100	144	100	64	₵
010 0101	045	37	25	%	100 0101	105	69	45	E	110 0101	145	101	65	€
010 0110	046	30	26	&	100 0110	106	70	46	Γ	110 0110	143	102	66	f
010 0111	047	33	27	'	100 0111	107	71	47	Ğ	110 0111	147	103	67	£
010 1C00	051	41	28	(	100 1000	110	72	48	H	110 1C00	151	104	F8	₼
010 1C01	051	41	29	)	100 1001	111	73	49	I	110 1C01	151	105	E9	i
010 1C10	052	42	2A	*	100 1010	112	74	4A	J	110 1C10	152	106	EA	j
010 1C11	053	43	2B	+	100 1011	113	75	4B	K	110 1C11	153	107	EB	k
010 1100	054	44	2C	,	100 1100	114	76	4C	L	110 1100	154	108	6C	l
010 1101	055	45	2D	-	100 1101	115	77	4D	M	110 1101	155	109	6D	m
010 1110	056	43	2E	.	100 1110	116	78	4E	N	110 1110	150	110	EE	r
010 1111	057	47	2F	/	100 1111	117	79	4F	O	110 1111	157	111	EΓ	c
011 0C00	060	43	30	C	101 0000	120	80	50	P	111 0C00	160	112	70	p
011 0C01	061	43	31	1	101 0001	121	81	51	Q	111 0C01	161	113	71	ç
011 0C10	062	51	32	2	101 0010	122	82	52	R	111 0C10	162	114	72	r
011 0C11	063	51	33	3	101 0011	123	83	53	S	111 0C11	163	115	73	€
011 0100	064	52	34	4	101 0100	124	84	54	T	111 0100	164	116	74	t
011 0U1U	065	53	35	5	101 0U1U	125	85	55	U	111 0U1U	165	117	75	₼
011 0110	066	54	36	6	101 0110	126	86	56	V	111 0110	165	118	76	v
011 0111	067	55	37	7	101 0111	127	87	57	W	111 0111	167	119	77	w
011 1C00	070	53	30	C	101 1000	130	90	50	X	111 1C00	170	120	70	x
011 1C01	071	57	39	€	101 1001	131	99	59	Y	111 1C01	171	121	79	γ
011 1C10	072	53	3A	:	101 1010	132	90	5A	Z	111 1C10	172	122	7A	z
011 1C11	073	53	3B	:	101 1011	133	91	5B	[	111 1C11	173	123	7B	{
011 1100	074	60	3C	=	101 1100	134	92	5C	\	111 1100	174	124	7C	
011 1101	075	61	3D	=	101 1101	135	93	5D	l	111 1101	175	125	7D	l
011 1110	076	62	3E	>	101 1110	136	94	5E	^	111 1110	173	126	7E	~
011 1111	077	63	3F	?	101 1111	137	95	5F	-					

# STRING SEBAGAI ARRAY

**String** pada python dapat dipandang sebagai array. **String** dalam Python adalah **array byte** yang mewakili karakter **unicode**.

L	A	T	I	H	A	N
---	---	---	---	---	---	---

0	1	2	3	4	5	6
---	---	---	---	---	---	---

```
a = 'LATIHAN'  
print(a[0])
```

```
>> L
```



# PANJANG DAN PENUGASAN INDEKS

Pengecekan panjang string dapat dilakukan dengan Fungsi berikut

```
a = 'LATIHAN'  
print(len(a))
```

```
>> 7
```

```
a = 'LATIHAN'  
print(a[0:4])
```

```
>> LATI
```

```
a = 'LATIHAN'  
print('LATIHAN' [0] , 'LATIHAN' [1] , 'LATIHAN' [2])
```

```
>> L A T
```



# INDEKS NEGATIF

Penulisan indeks negatif pada python dapat dilakukan berdasarkan posisi dari kanan **string**.

L	A	T	I	H	A	N
---	---	---	---	---	---	---

0	1	2	3	4	5	6
-7	-6	-5	-4	-3	-2	-1

```
a = 'LATIHAN'  
print(a[-5])
```

```
>> T
```



# PENUGASAN INDEKS NEGATIF

```
a = 'LATIHAN'  
print(a[-6:-3])
```

```
>> ATI
```

```
a = 'LATIHAN'  
print(a[-4:7])
```

```
>> IHAN
```

```
a = 'LATIHAN'  
print('LATIHAN' [-3], 'LATIHAN' [-2], 'LATIHAN' [-1])
```

```
>> H A N
```



# MENGINDEKS DAN MENGIRIS BATAS

Penulisan indeks batas dapat dilakukan dengan penulisan default dengan menghilangkan nilai batasnya.

```
a = 'LATIHAN'  
print(a[:4])
```

```
>> LATI
```

```
a = 'LATIHAN'  
print(a[2:])
```

```
>> TIHAN
```



```
a = 'LATIHAN'  
print(a[:-2])
```

```
>> LATIH
```

```
a = 'LATIHAN'  
print(a[-3:])
```

```
>> HAN
```

```
a = 'LATIHAN'  
print('LATIHAN' [:2], 'LATIHAN' [-3:])
```

```
>> LA HAN
```



# MENGIRIS BATAS

Python tidak mengizinkan untuk menuliskan **indeks** di luar batas, akan tetapi dapat dilakukan untuk mengiris batas.

```
a = 'LATIHAN'  
print(a[0:6])
```

```
>> LATIHA
```

```
a = 'LATIHAN'  
print(a[0:10])
```

```
>> LATIHAN
```



# OPERATOR STRING

Masih ingat **membership**?

```
a = 'Institut Teknologi Bacharuddin Jusuf Habibie'  
print('Jusuf' in a)
```

```
>> True
```

```
a = 'Institut Teknologi Bacharuddin Jusuf Habibie'  
print('habibie' not in a)
```

```
>> True
```



# OPERATOR STRING 2

Masih ingat Operator **Identitas**?

```
a = 'ITH'  
print('ITH' is a)
```

```
>> True
```

```
a = 'ITH'  
print('ith' is not a)
```

```
>> True
```



# PENGGABUNGAN DAN PERULANGAN PADA STRING

Beberapa string dapat digabung menjadi string yang bersatu. Penulisan-nya dilakukan dengan memberi operator plus “+”

```
a = 'Institut'  
b = 'Teknologi'  
c = 'Bacharuddin'  
d = 'Jusuf'  
e = 'Habibie'  
ith = a+b+c+d+e  
print(ith)
```

```
>> InstitutTeknologiBacharuddinJusufHabibie
```

```
ith = a + ' ' + b + ' ' + c + ' ' + d + ' ' + e  
print(ith)
```

```
>> Institut Teknologi Bacharuddin Jusuf Habibie
```



# PENGULANGAN STRING

- Pengulangan string dapat dilakukan tanpa menuliskan manual secara berulang.
- Penulisan string berulang dilakukan dengan penulisan (\*) dilanjutkan dengan nilai jumlah perulangannya.

```
print('LATIHAN' * 4)
```

```
>> LATIHANLATIHANLATIHANLATIHAN
```



# METODE STRING

Metode string adalah proses yang melakukan tugas pada string. Bentuk umum dari ekspresi yang menerapkan metode adalah:

**stringName . methodName ()**



# METODE COUNT

Mengembalikan berapa kali nilai tertentu muncul dalam string.

```
string.count(value,start,end)
```

Parameter	Deskripsi
value	Nilai yang akan dicari
start	Opsional. integer posisi mulai pencarian, default 0
end	Opsional. integer posisi akhir pencarian, default len(string)



## COUNT 2

Beberapa contoh metode count:

```
str1 = 'santapan kita setiap jam setengah satu  
siang satu soto sapi sama seratus tusuk sate  
sapi pula'  
a=str1.count('sapi')  
print(a)
```

```
>> 2
```

```
a=str1.count('sapi',70,93)  
print(a)
```

```
>> 1
```



# METODE UPPER

Mengembalikan semua karakter menjadi kapital, simbol dan Angka akan diabaikan

```
string.upper()
```

```
str1 = 'latihan PRAKTIKUM'  
a=str1.upper()  
print(a)
```

```
>> LATIHAN PRAKTIKUM
```



# METODE LOWER

Mengembalikan semua karakter menjadi huruf kecil, simbol dan Angka akan diabaikan

```
string.lower()
```

```
str1 = 'latihan PRAKTIKUM'  
a=str1.lower()  
print(a)
```

```
>> latihan praktikum
```



# METODE IS-UPPER

Mengembalikan nilai **True** apabila semua karakter merupakan kapital, simbol dan Angka akan diabaikan. Sebaliknya **False**

```
string.isupper()
```

```
str1 = 'LATIHAN PRAKTIKUM'  
a=str1.isupper()  
print(a)
```

```
>> True
```



# METODE IS-LOWER

Mengembalikan nilai **True** apabila semua karakter merupakan huruf kecil, simbol dan Angka akan diabaikan. Sebaliknya **False**

```
string.islower()
```

```
str1 = 'latihan praktikum'  
a=str1.islower()  
print(a)
```

```
>> True
```



## METODE TITLE

Mengubah huruf pertama string menjadi huruf besar dan huruf setelahnya menjadi huruf kecil **untuk setiap kata**. Jika ada simbol atau Angka maka karakter setelah itu akan dibuat menjadi huruf besar.

```
string.title()
```

```
str1 = 'latiHAN prAKTIkum'  
a=str1.title()  
print(a)
```

```
>> Latihan Praktikum
```



# METODE IS-TITLE

Mengembalikan **True** apabila dalam string huruf pertama string merupakan huruf besar dan huruf setelahnya merupakan huruf kecil untuk setiap kata. Jika ada simbol atau Angka maka karakter setelah itu akan dicek apakah merupakan huruf besar. Sebaliknya maka **False**.

```
string.istitle()
```

```
str1 = 'latiHAN prakTIkum'  
a=str1.istitle()  
print(a)
```

```
>> False
```



# METODE CAPITALIZE

Dalam satu string mengubah huruf pertama string menjadi huruf besar dan setelahnya menjadi huruf kecil. Jika ada simbol atau Angka maka diabaikan.

```
string.capitalize()
```

```
str1 = 'latiHAN prAKTIkum'  
a=str1.capitalize()  
print(a)
```

```
>> Latihan praktikum
```



# METODE CASEFOLD

Dalam satu string mengubah semua huruf string menjadi huruf kecil. Jika ada simbol atau Angka maka diabaikan.

```
string.casefold()
```

```
str1 = 'latiHAN prakTIkum'  
a=str1.casefold()  
print(a)
```

```
>> latihan praktikum
```



# METODE STARTSWITH

Mengembalikan **True** jika string dimulai dengan nilai yang ditentukan.  
**False** jika sebaliknya.

```
string.startswith(value,start,end)
```

Parameter	Deskripsi
value	Nilai yang akan diperiksa apakah dimulai dengan nilai tersebut.
start	Opsional. integer posisi mulai pencarian, default 0
end	Opsional. integer posisi akhir pencarian, default len(string)



## STARTSWITH 2

Beberapa contoh metode startswith:

```
str1 = 'hallo, selamat bergabung'  
a=str1.startswith('hallo')  
print(a)
```

```
>> True
```

```
a=str1.startswith('selamat',7,15)  
print(a)
```

```
>> True
```



# METODE ENDSWITH

Mengembalikan **True** jika string diakhiri dengan nilai yang ditentukan.  
**False** jika sebaliknya.

```
string.endswith(value, start, end)
```

Parameter	Deskripsi
value	Nilai yang akan diperiksa apakah diakhiri dengan nilai tersebut.
start	Opsional. integer posisi mulai pencarian, default 0
end	Opsional. integer posisi akhir pencarian, default len(string)



## ENDSWITH 2

Beberapa contoh metode endswith:

```
str1 = 'hallo, selamat bergabung'  
a=str1.endswith('bung')  
print(a)
```

```
>> True
```

```
a=str1.endswith('amat',7,14)  
print(a)
```

```
>> True
```



# METODE JOIN

Mengambil semua item dalam urutan dan menggabungkannya dalam satu string. Memakai join **harus** ditentukan pemisahnya.

```
string.join(urutan)
```

Parameter	Deskripsi
Urutan	Objek yang semua nilainya akan diubah menjadi string



## JOIN 2

Beberapa contoh metode join:

```
tuple1 = ('Kota', 'Parepare')
a = '#' . join(tuple1)
print(a)
```

```
>> Kota#Parepare
```

```
dict1 = { 'Kampus' : 'ITH' , 'Kota' : 'Parepare' }
a = '$$' . join(dict1)
print(a)
```

```
>> Kampus$$Kota
```



# METODE SPLIT

Membagi string menjadi daftar. Dapat ditentukan jenis pemisahnya dan pemisah defaultnya spasi.

```
string.split(separator,maxsplit)
```

Parameter	Deskripsi
Separator	Opsional, pemisah yang akan digunakan saat memisahkan string. Secara default spasi putih apa pun adalah pemisah
Maxsplit	Opsional, jumlah split yang akan dilakukan, nilai default adalah -1 (semua kejadian)



## SPLIT 2

Beberapa contoh metode split:

```
str1 = "halo selamat datang"  
a = str1.split()  
print(a)
```

```
>> ['halo', ' selamat', ' datang']
```

```
str1 = "halo# selamat datang"  
a = str1.split('#')  
print(a)
```

```
>> ['halo', ' selamat datang']
```



## SPLIT 3

Beberapa contoh metode split:

```
str1 = "halowkselamatwkdatang"  
a = str1.split('wk')  
print(a)
```

```
>> [ 'halo' , ' selamat' , ' datang' ]
```

```
str1 = "halowkselamatwkdatang"  
a = str1.split('wk',1)  
print(a)
```

```
>>[ 'halo' , ' selamatwkdatang' ]
```



# METODE REPLACE

Mengganti suatu string tertentu dengan string tertentu yang ditentukan.

```
string.replace(oldvalue, newvalue, count)
```

Parameter	Deskripsi
oldvalue	Diharuskan. String untuk diganti
newvalue	Diharuskan. String pengganti
Count	Opsional. Jumlah kemunculan value yang akan diganti. Defaultnya adalah semua kemunculan



## REPLACE 2

Beberapa contoh metode replace:

```
str1 = "halo selamat datang"  
a = str1.replace('datang','bergabung')  
print(a)
```

```
>> halo selamat bergabung
```

```
str1 = "one was a race horse, two two was one too"  
a = str1.replace('one','two')  
print(a)
```

```
>>two was a race horse, two two was two too
```



## REPLACE 3

```
str1 = "two was a race horse, two two was two too"  
a = str1.replace('two','three',2)  
print(a)
```

```
>>three was a race horse, three two was two too
```



# METODE FIND

Metode untuk Mencari karakter berada di posisi berapa paling awal. Jika karakter yang dicari tidak ada, maka komputer akan mengembalikan nilai **-1**.

```
string.find(value, start, end)
```

Parameter	Deskripsi
value	Diharuskan. Nilai yang akan dicari dari karakter paling kiri.
start	Opsional. integer posisi mulai pencarian, default 0
end	Opsional. integer posisi akhir pencarian, default len(string)



## FIND 2

Beberapa contoh metode find:

```
str1 = "cari-tanda-minus-pada-kalimat"  
a = str1.find('-')  
print(a)
```

```
>> 4
```

```
str1 = "cari-tanda-minus-pada-kalimat"  
a = str1.find('-',5,12)  
print(a)
```

```
>> 10
```



# METODE R-FIND

Metode untuk Mencari karakter berada di posisi berapa paling terakhir. Jika karakter yang dicari tidak ada, maka komputer akan mengembalikan nilai **-1**.

```
string.rfind(value,start,end)
```

Parameter	Deskripsi
value	Diharuskan. Nilai yang akan dicari dari karakter paling kanan. Atau posisi paling terakhir
start	Opsional. integer posisi mulai pencarian, default 0
end	Opsional. integer posisi akhir pencarian, default len(string)



## R-FIND 2

Beberapa contoh metode rfind:

```
str1 = "cari-tanda-minus-pada-kalimat"  
a = str1.rfind('-')  
print(a)
```

```
>> 21
```

```
str1 = "cari-tanda-minus-pada-kalimat"  
a = str1.rfind('-',0,12)  
print(a)
```

```
>> 10
```



# METODE R-JUST

Menggeser string ke kanan menggunakan karakter tertentu sebagai isian space, spasi merupakan default dari isian.

**string.rjust(length, character)**

Parameter	Deskripsi
Length	Diharuskan, panjang string yang dikembalikan
Character	Opsional, sebuah karakter untuk mengisi ruang yang hilang (di sebelah kiri string). Standarnya adalah " " (spasi).



## R-JUST 2

Beberapa contoh metode rjust:

```
str1 = "KANAN"  
a = str1.rjust(10)  
print(a)
```

```
>> KANAN
```

```
str1 = "KANAN"  
a = str1.rjust(10, 'x')  
print(a)
```

```
>>XXXXXKANAN
```



# METODE L-JUST

Menggeser string ke kiri menggunakan karakter tertentu sebagai isian, spasi merupakan default dari isian.

```
string.ljust(length, character)
```

Parameter	Deskripsi
Length	Diharuskan, panjang string yang dikembalikan
Character	Opsional, sebuah karakter untuk mengisi ruang yang hilang (di sebelah kanan string). Standarnya adalah " " (spasi).



## L-JUST 2

Beberapa contoh metode ljust:

```
str1 = "KIRI"  
a = str1.ljust(10)  
print(a)
```

```
>> KIRI
```

```
str1 = "KIRI"  
a = str1.ljust(10, 'x')  
print(a)
```

```
>> KIRIXXXXXX
```



# METODE CENTER

Mengatur agar string berada ditengah, dengan memberikan isian di kiri dan kanan. Spasi merupakan default dari isian.

## **string.center(length, character)**

Parameter	Deskripsi
Length	Diharuskan, panjang string kiri dan kanan yang dikembalikan
Character	Opsional, sebuah karakter untuk mengisi ruang yang hilang (di sebelah kiri dan kanan string). Standarnya adalah " " (spasi).



## CENTER 2

Beberapa contoh metode center:

```
str1 = "TENGAH"  
a = str1.center(10)  
print(a)
```

```
>> TENGAH
```

```
str1 = "TENGAH"  
a = str1.center(10, 'X')  
print(a)
```

```
>> XXTENGAHXX
```



# METODE STRIP

Menghilangkan karakter Awal (spasi di awal) dan karakter spasi akhir (spasi di akhir). Spasi merupakan karakter utama yang dihapus.

## **string.strip(character)**

Parameter	Deskripsi
Character	Opsional, Satu set karakter untuk dihapus sebagai karakter utama/ pengikut.



## STRIP 2

Beberapa contoh metode strip:

```
str1 = " 3kiri-3kanan "
a = str1.strip()
print(a)
```

```
>> 3kiri-3kanan
```

```
str1 = ",,x,:haloo semua::xx::.,x,"
a = str1.strip(',x:.')
print(a)
```

```
>> haloo semua
```



# FORMAT STRING LANJUTAN

Menggabungkan string dengan format lain seperti angka, sebelumnya dapat digunakan dengan "f", namun dengan format() juga dapat digunakan. Dengan memasang placeholder {} pada string maka isian format(isian) akan ditempatkan ke placeholder.

```
masuk = 'angkatan {}'  
tahun = 2023  
a = masuk.format(tahun)  
print(a)
```

```
>> angkatan 2023
```



# FORMAT STRING LANJUTAN 2

```
data_saya = 'angkatan {} prodi {} semester {}'  
tahun = 2023  
prodi = 'Sistem Informasi'  
semester = 1  
a = data_saya.format(tahun,prodi,semester)  
print(a)
```

```
>> angkatan 2023 prodi Sistem Informasi semester 1
```



## STRING MULTILINE

Pembuatan string dapat dibuat berbaris-baris sesuai keinginan. Penulisannya sebagai Berikut.

```
str1 = """Semoga di masa yang akan datang ITH  
menghasilkan Habibie-Habibie baru yang akan  
mensejahterakan masyarakat melalui pengembangan  
IPTEKS"""\n\nprint(a)
```

```
>> Semoga di masa yang akan datang ITH  
menghasilkan Habibie-Habibie baru yang akan  
mensejahterakan masyarakat melalui pengembangan  
IPTEKS
```



# INPUT STRING

Seluruh input Tanpa properti akan dibaca sebagai string.

```
str1 = input('Masukkan apa saja =')
print(str1,'tipenya=',type(str1))
```

Apabila yang dimasukkan adalah 220, maka hasil nya adalah string 220

```
>> 220 tipenya= <class 'str'>
```



## REFERENCES

<https://www.bhutanpythoncoders.com>

<https://www.w3schools.com/>

