CS310 COMPUTER PROGRAMMING STRING

School of Information Technology and Innovation

String

ข้อมูลสตริง (String) คือ ตัวอักขระ ข้อความหรือประโยคที่ประกอบด้วยตัวอักขระหลายๆ ตัวมาต่อกัน โดยเก็บข้อมูลอยู่ในเครื่องหมาย “....” หรือ ‘.....’ ซึ่งมีจัดเก็บสตริงแบบ Unicode ทำให้รองรับการจัดเก็บภาษาทั่วโลกได้

Assigning a string to a variable is done with the variable name followed by an equal sign and display a string literal with the print() function:

```
str1 = "Python Program"
str2 = 'Python Program'
print(str1)
print(str2)
```

Result

```
Python Program
Python Program
```

Converting with Strings

We can convert numbers to strings through using the str() method. We'll pass either a number or a variable into the parentheses of the method and then that numeric value will be converted into a string value.

```
num = 245
str1 = str(num)
print(str1)
print("Type variable str1 is ",type(str1))
print("Type variable num is ",type(num))
```

Result

```
245
Type variable str1 is  <class 'str'>
Type variable num is  <class 'int'>
```

Converting with Strings

หลังจากมีการแปลงชนิดข้อมูลจำนวนเต็มเป็นชนิดสตริงแล้ว และนำตัวดำเนินการคณิตศาสตร์มาดำเนินการกับชนิดข้อมูลทั้งสอง จะทำให้เกิดการแจ้งเตือนข้อผิดพลาด

```
num = 347
num1 = 245
str1 = str(num)
result = str1 + num
print(result)
```

Result

```
TypeError                 Traceback (most recent call last)
<ipython-input-2-f793c5b40c68> in <module>()
      2 num1 = 245
      3 str1 = str(num)
----> 4 result = str1 + num
      5 print(result)
TypeError: must be str, not int
```

Concatenating with Strings

To join string data types together Variables can be concatenated with (....) And separated by comma (,) or operators (+) connect strings together and can use ("...") spaces between text.

```
str1 = "Hello"
str2 = "Python"
str3 = "Programming"
print(str1, str2, str3)
print(str1 + " " + str2 + " " + str3)
```

Result

```
Hello Python Programming
Hello Python Programming
```

Multiply with Strings

The string can be repeated by using the operator (*). Specify the number of times to repeat.

We can also multiply a string to repeat it:

```
str1 = "Python "
str2 = "Python"
print(str1 * 3)
print(3 * str1)
print(3 * str2)
```

Result

```
Python Python Python
Python Python Python
PythonPythonPython
```

```
print(40 * "-")
print(40 * "*")
```

Result

```
----------------------------------------
****************************************
```

Length with Strings

len() function is an inbuilt function in Python programming language that returns the length of the string.

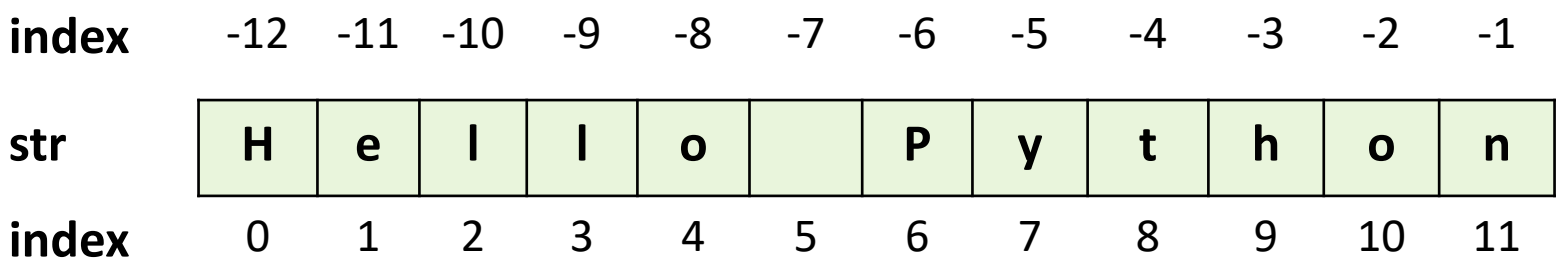
```
str1 = "Python"
str2 = "python "
str3 = ""
str4 = " "
str5 = "character or string"
print("amount of characters in str1 = ",len(str1))
print("amount of characters in str2 = ",len(str2))
print("amount of characters in str3 = ",len(str3))
print("amount of characters in str4 = ",len(str4))
print("amount of characters in str5 = ",len(str5))
```

Result

```
amount of characters in str1 =  6
amount of characters in str2 =  7
amount of characters in str3 =  0
amount of characters in str4 =  1
amount of characters in str5 =  19
```

การเข้าถึงตำแหน่งตัวอักขระในชนิดข้อมูลสตริง (Indexing String Operator)

In Python, strings are ordered sequences of character data, and thus can be indexed in this way. Individual characters in a string can be accessed by specifying the string name followed by a number in square brackets [ ]. the first character in the string has index 0, or refers to the last character has index -1



การเข้าถึงตำแหน่งตัวอักขระในชนิดข้อมูลสตริง (Indexing String Operator)

The individual characters can be accessed by index as follows:

```
str1 = "Hello Python"
print("amount of characters in str1 = ",len(str1))
print("str1 index 0 is ", str1[0])
print("str1 index -1 is ", str1[-1])
print("str1 index 5 is ", str1[5])
print("str1 index -5 is ", str1[-5])
```

Result

```
amount of characters in str1 =  12
str1 index 0 is  H
str1 index -1 is  n
str1 index 5 is
str1 index -5 is  y
```

Slicing with Strings

You can return a range of characters by using the slice syntax.

Specify the start index and the end index, separated by a colon, to return a part of the string.

```
str1 = "Hello Python"
print("String slicing 0 : 4 result ", str1[0:4])
print("String slicing -6 : -1 result ", str1[-6:-1])
print("String slicing 6 : 12 result ", str1[6:12])
print("String slicing 0 : 5 result ", str1[0:5])
print("String slicing result ", str1[:])
```

Result

```
String slicing 0 : 4 result  Hell
String slicing -6 : -1 result  Pytho
String slicing 6 : 12 result  Python
String slicing 0 : 5 result  Hello
String slicing result  Hello Python
```

Multiline with Strings

You can assign a multiline string to a variable by using three quotes:

```
str = """Python Programming,
Python Programming,
Python Programming."""
print(str)
```

Result

```
Python Programming,
Python Programming,
Python Programming.
```

Escape Sequence of Python String

In Python strings, the backslash "\" is a special character, also called the "escape" character. It is used in representing certain whitespace characters: "\t" is a tab, "\n" is a newline, and "\r" is a carriage return.

Escape Sequence	Description
\newline	Backslash and newline ignored
\\	Backslash (\) ใส่เครื่องหมาย \
\'	Single quote (‘) ใส่เครื่องหมาย ‘
\"	Double quote (") ใส่เครื่องหมาย “
\a	ASCII Bell (BEL) ให้ส่งเสียงเตือน
\b	ASCII Backspace (BS) ให้เลื่อนเคอร์เซอร์ถอยหลังไป 1 ตำแหน่ง
\f	ASCII Formfeed (FF) ให้ขึ้นหน้าใหม่
\n	ASCII Linefeed (LF) ให้ขึ้นบรรทัดใหม่
\r	ASCII Carriage Return (CR) ให้เคอร์เซอร์อยู่ทางซ้าย
\t	ASCII Horizontal Tab (TAB) ให้แสดงแท็บตามแนวนอน
\v	ASCII Vertical Tab (VT) ให้แสดงแท็บตามแนวตั้ง

String Format

As we learned in the Python Variables chapter, we cannot combine strings and numbers like this:

```
age = 36
txt = "My name is John, I am " + age
print(txt)
```

Result

```
TypeError       Traceback (most recent call last)
<ipython-input-16-4d1b227cff41> in <module>()
      1 age = 36
----> 2 txt = "My name is John, I am " + age
      3 print(txt)
TypeError: must be str, not int
```

String Format

But we can combine strings and numbers by using the format() method!

The format() method takes the passed arguments, formats them, and places them in the string where the placeholders {} are:

```
age = 36
txt = "My name is John, and I am {}"
print(txt.format(age))
```

Result

```
My name is John, and I am 36
```

String Format

The format() method takes unlimited number of arguments, and are placed into the respective placeholders:

```
quantity = 3
itemno = 567
price = 49.95
myorder = "I want {} pieces of item {} for {} dollars."
print(myorder.format(quantity, itemno, price))
```

Result

```
I want 3 pieces of item 567 for 49.95 dollars.
```

String Format

You can use index numbers {0} to be sure the arguments are placed in the correct placeholders:

```
quantity = 3
itemno = 567
price = 49.95
myorder = "I want to pay {2} dollars for {0} pieces of item {1}."
print(myorder.format(quantity, itemno, price))
```

Result

```
I want to pay 49.95 dollars for 3 pieces of item 567.
```

How to Reverse a String in Python

There is no built-in function to reverse a String in Python.

The fastest (and easiest?) way is to use a slice that steps backwards, -1.

Example : Reverse the string "Hello World":

```
txt = "Hello World" [ : :-1]
print(txt)
```

Result
dlroW olleH

String Methods

Python has a set of built-in methods that you can use on strings.

Note: All string methods returns new values. They do not change the original string.

Method	Description
capitalize()	Converts the first character to upper case
center()	Returns a centered string
count()	Returns the number of times a specified value occurs in a string
find()	Searches the string for a specified value and returns the position of where it was found

Method	Description
format()	Formats specified values in a string
index()	Searches the string for a specified value and returns the position of where it was found
isalnum()	Returns True if all characters in the string are alphanumeric
isdigit()	Returns True if all characters in the string are digits
islower()	Returns True if all characters in the string are lower case
isnumeric()	Returns True if all characters in the string are numeric
isspace()	Returns True if all characters in the string are whitespaces
isupper()	Returns True if all characters in the string are upper case
join()	Joins the elements of an iterable to the end of the string
ljust()	Returns a left justified version of the string
lower()	Converts a string into lower case
lstrip()	Returns a left trim version of the string

Method	Description
partition()	Returns a tuple where the string is parted into three parts
replace()	Returns a string where a specified value is replaced with a specified value
rjust()	Returns a right justified version of the string
rsplit()	Splits the string at the specified separator, and returns a list
rstrip()	Returns a right trim version of the string
split()	Splits the string at the specified separator, and returns a list
swapcase()	Swaps cases, lower case becomes upper case and vice versa
upper()	Converts a string into upper case

สามารถดูข้อมูล Method อื่นๆ เพิ่มเติมได้ที่ https://www.w3schools.com/python/python_strings.asp

Python String capitalize() Method

The capitalize() method returns a string where the first character is upper case.

string.capitalize()

Example : Upper case the first letter in this sentence:

```
txt = "hello, and welcome to my world."
x = txt.capitalize()
print (x)
```

Result

```
Hello, and welcome to my world.
```

Example : See what happens if the first character is a number:

```
txt = "36 is my age."
x = txt.capitalize()
print (x)
```

Result

```
36 is my age.
```

Python String center() Method

The center() method will center align the string, using a specified character (space is default) as the fill character.

string.center(*length, character*)

length	Required. The length of the returned string
character	Optional. The character to fill the missing space on each side. Default is " " (space)

Example : Print the word "banana", taking up the space of 20 characters, with "banana" in the middle:

```
txt = "banana"
x = txt.center(20)
print(x)
```

Result

^^^^^^^banana^^^^^^^

Note: ^ represents space

Example : Using the letter "O" as the padding character:

```
txt = "banana"
x = txt.center(20, "O")
print(x)
```

Result

OOOOOOObananaOOOOOOO

Python String count() Method

The count() method returns the number of times a specified value appears in the string.

string.count(value, start, end)

value	*Required. A String. The string to value to search for*
start	*Optional. An Integer. The position to start the search. Default is 0*
end	*Optional. An Integer. The position to end the search. Default is the end of the string*

Example : Return the number of times the value "apple" appears in the string:

```
txt = "I love apples, apple are my favorite fruit"
x = txt.count("apple")
print(x)
```

Result
2

Example : Search from position 10 to 24:

```
txt = "I love apples, apple are my favorite fruit"
x = txt.count("apple", 10, 24)
print(x)
```

Result
1

Python String find() Method

The find() method finds the first occurrence of the specified value.

The find() method returns -1 if the value is not found.

The find() method is almost the same as the index() method, the only difference is that the index() method raises an exception if the value is not found. (See example below)

string.find(value, start, end)

value	*Required. The value to search for*
start	*Optional. Where to start the search. Default is 0*
end	*Optional. Where to end the search. Default is to the end of the string*

Example : Where in the text is the word "welcome"?:

```
txt = "Hello, welcome to my world."
x = txt.find("welcome")
print(x)
```

Result
7

Python String find() Method

Example : Where in the text is the first occurrence of the letter "e"?:

```
txt = "Hello, welcome to my world."
x = txt.find("e")
print(x)
```

Result
1

Example : Where in the text is the first occurrence of the letter "e" when you only search between position 5 and 10?:

```
txt = "Hello, welcome to my world."
x = txt.find("e", 5, 10)
print(x)
```

Result
8

Python String find() Method

Example : If the value is not found, the find() method returns -1, but the index() method will raise an exception:

```
txt = "Hello, welcome to my world."
print(txt.find("q"))
print(txt.index("q"))
```

Result

```
-1
Traceback (most recent call last):
  File "demo_ref_string_find_vs_index.py", line 4 in <module>
    print(txt.index("q"))
ValueError: substring not found
```

Python String index() Method

The index() method finds the first occurrence of the specified value.

The index() method raises an exception if the value is not found.

The index() method is almost the same as the find() method, the only difference is that the find() method returns -1 if the value is not found. (See example below)

string.index(value, start, end)

value	*Required. The value to search for*
start	*Optional. Where to start the search. Default is 0*
end	*Optional. Where to end the search. Default is to the end of the string*

Example : Where in the text is the word "welcome"?:

```
txt = "Hello, welcome to my world."
x = txt.index("welcome")
print(x)
```

Result
7

Python String index() Method

Example : Where in the text is the first occurrence of the letter "e"?:

```
txt = "Hello, welcome to my world."
x = txt.index("e")
print(x)
```

Result
1

Example : Where in the text is the first occurrence of the letter "e" when you only search between position 5 and 10?:

```
txt = "Hello, welcome to my world."
x = txt.index("e", 5, 10)
print(x)
```

Result
8

Python String index() Method

Example : If the value is not found, the find() method returns -1, but the index() method will raise an exception:

```
txt = "Hello, welcome to my world."
print(txt.find("q"))
print(txt.index("q"))
```

Result

```
-1
Traceback (most recent call last):
  File "demo_ref_string_find_vs_index.py", line 4 in <module>
    print(txt.index("q"))
ValueError: substring not found
```

Python String isalnum() Method

The isalnum() method returns True if all the characters are alphanumeric, meaning alphabet letter (a-z) and numbers (0-9).

Example of characters that are not alphanumeric: (space)!#%&? etc.

string.isalnum( )

Example : Check if all the characters in the text are alphanumeric:

```
txt = "Company12"
x = txt.isalnum()
print(x)
```

Result

```
True
```

```
txt = "Company 12"
x = txt.isalnum()
print(x)
```

Result

```
False
```

Python String isdecimal() Method

The isdecimal() method returns True if all the characters are decimals (0-9).This method is used on unicode objects.

string.isdecimal()

Example : Check if all the characters in the unicode object are decimals:

```
txt = "\u0033" #unicode for 3
txt1 = "1234"
txt2 = "123 456"
txt3 = "345 Python"
print(txt.isdecimal())
print(txt1.isdecimal())
print(txt2.isdecimal())
print(txt3.isdecimal())
```

Result

```
True
True
False
False
```

Python String isdigit() Method

The isdigit() method returns True if all the characters are digits, otherwise False. Exponents, like ², are also considered to be a digit.

string.isdigit()

Example : Check if all the characters in the are digits:

```
txt = "50800"
x = txt.isdigit()
print(x)
```

Result

```
True
```

```
a = "\u0030" #unicode for 0
b = "\u00B2" #unicode for ²

print(a.isdigit())
print(b.isdigit())
```

Result

```
True
True
```

Python String islower() Method

The islower() method returns True if all the characters are in lower case, otherwise False. Numbers, symbols and spaces are not checked, only alphabet characters.

string.islower()

Example : Check if all the characters in the text are in lower case:

```
txt = "hello world!"
a = "Hello world!"
b = "hello 123"
c = "mynameisPeter"
print(txt.islower())
print(a.islower())
print(b.islower())
print(c.islower())
```

Result

```
True
False
True
False
```

Python String isspace() Method

The isspace() method returns True if all the characters in a string are whitespaces, otherwise False.

string.isspace()

Example : Check if all the characters in the text are whitespaces:

```
txt1 = "    "
txt2 = "    s    "
txt3 = ""
print(txt1.isspace())
print(txt2.isspace())
print(txt3.isspace())
```

Result

```
True
False
False
```

Python String join() Method

The join() method takes all items in an iterable and joins them into one string. A string must be specified as the separator.

string.join(iterable)

Example : Join all items in a tuple into a string, using a hash character as separator:

```
myTuple = ("John", "Peter", "Vicky")
print("#".join(myTuple))
list1 = ['g','e','e','k', 's']
print("".join(list1))
```

Result

```
John#Peter#Vicky
geeks
```

Python String isupper() Method

The isupper() method returns True if all the characters are in upper case, otherwise False. Numbers, symbols and spaces are not checked, only alphabet characters.

string.isupper()

Example : Check if all the characters in the text are in upper case:

```
txt1 = "THIS IS NOW!"
txt2 = "Python Programming"
txt3 = "123#$%"
print(txt1.isupper())
print(txt2.isupper())
print(txt3.isupper())
```

Result

```
True
False
False
```

Python String ljust() Method

The ljust() method will left align the string, using a specified character (space is default) as the fill character.

string.ljust(length, character)

length	Required. The length of the returned string
character	Optional. A character to fill the missing space (to the right of the string). Default is " " (space).

Example : Return a 20 characters long, left justified version of the word "banana":

```
txt = "banana"
print(txt.ljust(20),"is my favorite fruit.")
print(txt.ljust(20,"^"),"is my favorite fruit.")
```

Result

```
Banana               is my favorite fruit.
banana^^^^^^^^^^^^^^ is my favorite fruit.
```

Python String lower() Method

The lower() method returns a string where all characters are lower case.

Symbols and Numbers are ignored.

string.lower( )

Example : Lower case the string:

```
txt = "Hello my FRIENDS"
print(txt.lower())
```

Result

```
hello my friends
```

Python String lstrip() Method

The lstrip() method removes any leading characters (space is the default leading character to remove)

string.lstrip(characters)

characters	*Optional. A set of characters to remove as leading characters*

Example : Remove spaces to the left of the string:

```
txt = "     banana     "
x = txt.lstrip()
print("of all fruits", x, "is my favorite")
txt1 = "....#C Python Java..."
txt2 = ",,,,,ssaaww.....banana"
print(txt1.lstrip("#."))
print(txt2.lstrip(",.asw"))
```

Result

```
of all fruits banana      is my favorite
C Python Java...
banana
```

Python String partition() Method

The partition() method searches for a specified string, and splits the string into a tuple containing three elements.

The first element contains the part before the specified string.

The second element contains the specified string.

The third element contains the part after the string.

string.partition(value)

value	*Required. The string to search for*

Note: This method search for the *first* occurrence of the specified string.

Python String partition() Method

Example : Search for the word "bananas", and return a tuple with three elements:

1 - everything before the "match"

2 - the "match"

3 - everything after the "match"

```
txt = "I could eat bananas all day"
x = txt.partition("bananas")
print(x)
```

Result

```
('I could eat ', 'bananas', ' all day')
```

Python String partition() Method

Example : If the specified value is not found, the partition() method returns a tuple containing:

1 - the whole string,

2 - an empty string,

3 - an empty string:

```
txt = "I could eat bananas all day"
x = txt.partition("apples")
print(x)
```

Result

```
('I could eat bananas all day', '', '')
```

Python String replace() Method

The replace() method replaces a specified phrase with another specified phrase.

string.replace(oldvalue, newvalue, count)

oldvalue	*Required. The string to search for*
newvalue	*Required. The string to replace the old value with*
count	*Optional. A number specifying how many occurrences of the old value you want to replace. Default is all occurrences*

Example : Replace the word "bananas":

```
txt = "I like bananas"
x = txt.replace("bananas", "apples")
print(x)
```

Result

```
I like apples
```

Python String replace() Method

Example : Replace all occurrence of the word "one":

```
txt = "one one was a race horse, two two was one too."
x = txt.replace("one", "three")
print(x)
```

Result

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

Note: *All* occurrences of the specified phrase will be replaced, if nothing else is specified.

Example : Replace the two first occurrence of the word "one":

```
txt = "one one was a race horse, two two was one too."
x = txt.replace("one", "three", 2)
print(x)
```

Result

```
three three was a race horse, two two was one too.
```

Python String rjust() Method

The rjust() method will right align the string, using a specified character (space is default) as the fill character.

string.rjust(length, character)

length	*Required. The length of the returned string*
character	*Optional. A character to fill the missing space (to the left of the string). Default is " " (space).*

Example : Return a 20 characters long, right justified version of the word "banana":

```
txt = "banana"
print(txt.rjust(20), "is my favorite fruit.")
print(txt.rjust(20, "O"), "is my favorite fruit.")
```

Result

```
              banana is my favorite fruit.
OOOOOOOOOOOOOObanana is my favorite fruit.
```

Python String rstrip() Method

The rstrip() method removes any trailing characters (characters at the end a string), space is the default trailing character to remove.

string.rstrip(characters)

character	*Optional. A set of characters to remove as trailing characters*

Example : Remove spaces to the right of the string:

```
txt = "     banana     "
x = txt.rstrip()
print("of all fruits", x, "is my favorite")
```

Result

```
of all fruits      banana is my favorite
```

Example : Remove the trailing characters:

```
txt = "banana,,,,,ssaaww....."
x = txt.rstrip(".w")
print(x)
```

Result

```
banana,,,,,ssaa
```

Python String split() Method

The split() method splits a string into a list.

You can specify the separator, default separator is any whitespace.

string.split(separator, max)

separator	*Optional. Specifies the separator to use when splitting the string. Default value is a whitespace*
max	*Optional. Specifies how many splits to do. Default value is -1, which is "all occurrences"*

Note: When max is specified, the list will contain the specified number of elements plus one.

Example : Split a string into a list where each word is a list item:

```
txt = "welcome to the jungle"
x = txt.split()
print(x)
```

Result

```
['welcome', 'to', 'the', 'jungle']
```

Python String split() Method

Example : Split the string, using comma, followed by a space, as a separator:

```
txt = "hello, my name is Peter, I am 26 years old"
print(txt.split(", "))
```

Result

```
['hello', 'my name is Peter', 'I am 26 years old']
```

Example : Use a hash character as a separator:

```
txt = "apple#banana#cherry#orange"
print(txt.split("#"))
```

Result

```
['apple', 'banana', 'cherry', 'orange']
```

Example : Use a hash character as a separator:

```
txt = "apple#banana#cherry#orange"
# setting the max parameter to 1, will return a
list with 2 elements!
print(txt.split("#", 1))
```

Result

```
['apple', 'banana#cherry#orange']
```

Python String strip() Method

The strip() method removes any leading (spaces at the beginning) and trailing (spaces at the end) characters (space is the default leading character to remove)

string.strip(characters)

characters	*Optional. A set of characters to remove as leading/trailing characters*

Example : Remove spaces at the beginning and at the end of the string:

```
txt = "     banana     "
x = txt.strip()
print("of all fruits", x, "is my favorite")
txt1 = ",,,,,rrttgg.....banana....rrr"
print(txt1.strip(",r"))
```

Result

```
of all fruits banana is my favorite
ttgg.....banana....
```

Python String swapcase() Method

The swapcase() method returns a string where all the upper case letters are lower case and vice versa.

string.swapcase()

characters	*Optional. A set of characters to remove as leading/trailing characters*

Example : Make the lower case letters upper case and the upper case letters lower case:

```
txt = "Hello My Name Is PETER"
x = txt.swapcase()
print(x)
```

Result

```
hELLO mY nAME iS peter
```

Python String upper() Method

The upper() method returns a string where all characters are in upper case. Symbols and Numbers are ignored.

string.upper()

characters	*Optional. A set of characters to remove as leading/trailing characters*

Example : Make the lower case letters upper case and the upper case letters lower case:

```
txt = "Hello my friends"
x = txt.upper()
print(x)
```

Result

```
HELLO MY FRIENDS
```

การดำเนินการกับชนิดข้อมูลอักขระหรือสตริง (String Operation)

สามารถนำเอาตัวดำเนินการต่างๆ มาใช้กับชนิดข้อมูลอักขระหรือสตริงได้เหมือนกับชนิดข้อมูลอื่นๆ เช่น การเปรียบเทียบ การเชื่อม การตรวจสอบการมีอยู่หรือไม่มีอยู่

```
str1 = "Hello Python"
str2 = "Programming"
str3 = "Programming"
print("str1 < str2 ? = ",str1 < str2)
print("str1 > str2 ? = ",str1 > str2)
print("str1 >= str2 ? = ",str1 >= str2)
print("str1 != str2 ? = ",str1 != str2)
print("str2 == str3 ? = ",str2 == str3)
```

Result

```
str1 < str2 ? = True
str1 > str2 ? = False
str1 >= str2 ? = False
str1 != str2 ? = True
str2 == str3 ? = True
```

การดำเนินการกับชนิดข้อมูลอักขระหรือสตริง (String Operation)

สามารถใช้ตัวดำเนินการ in และ not in ตรวจสอบอักขระว่ามีหรือไม่มีในข้อความ ผลลัพธ์ที่ได้จะเป็น True หรือ False

```
str1 = "Hello Python Programming"
str2 = "Bangkok University"
print("character i in str1? = ","i" in str1)
print("character A in str2? = ","A" in str2)
print("character n not in str2? = ","n" not in str2)
```

Result

```
character i in str1? = True
character A in str2? = False
character n not in str2? = False
```

Assignment 1 : Split string in odd/even position

จงเขียนโปรแกรมรับค่าสตริง (String) จากแป้นพิมพ์ และเก็บในตัวแปร Text และให้สร้างรหัส (Encode) จากข้อมูลในตัวแปร Text โดยแบ่งข้อมูลเป็น 2 ชุด คือ Code1 (เก็บเฉพาะตัวอักษรที่อยู่ในตำแหน่งเลขคู่) และ Code2 (เก็บตัวอักษรที่เหลือ คือ ตัวอักษรในตำแหนงเลขคี่) พร้อมพิมพ์ผลลัพธ์เป็นรหัส 2 ชุดออกหน้าจอ โดยมีการรับและแสดงผลข้อมูลดังตัวอย่าง

ตัวอย่างการรับและแสดงผลข้อมูล

```
Please enter your code : HaNaga123Yoohoo
===process encode===
code1 = HNg13oho
code2 = aaa2Yoo
```

Assignment 2 : Email generator program

จงเขียนโปรแกรมสร้างอีเมลให้กับพนักบริษัท science.com โดยโปรแกรมจะรับชื่อและนามสกุลจากผู้ใช้ทางแป้นพิมพ์ และโปรแกรมจะทำการสร้างอีเมลให้พนักงาน โดยนำชื่อและนามสกุลตัวแรกของพนักงานมาสร้างเป็นอีเมล โดยมีการรับและแสดงผลข้อมูลดังตัวอย่าง

ตัวอย่างการรับและแสดงผลข้อมูล

```
Enter name and lastname : Thamaraj ketpong
Your email = Thamaraj.k@science.com
```

Assignment 3 : Convert letter

จงเขียนโปรแกรมรับข้อความมา 1 ข้อความ พร้อมรับตัวอักษรที่ต้องการนับภายในข้อความนั้น และแสดงผลจำนวนตัวอักษรที่ได้และ Convert ตัวอักษร และแสดงข้อความย้อนกลับ แสดงผลออกหน้าจอดังตัวอย่าง

ตัวอย่างการรับและแสดงผลข้อมูล

```
Enter sentence : Place HalF a poTato in The reFrigerator.
Enter character to count : i
amount i in sentence = 2

Convert to upper case
=====================
PLACE HALF A POTATO IN THE REFRIGERATOR.

Convert to lower case
=====================
place half a potato in the refrigerator.

Convert to swap case
=====================
pLACE hALf A POtATO IN tHE REfRIGERATOR.

Reverse sentence
=====================
.rotaregirFer ehT ni otaTop a FlaH ecalP
```

Assignment 4 : Short month

จงเขียนโปรแกรมรับค่าสตริง (String) เก็บในตัวแปรเป็นชื่อเดือน (month) ซึ่งรับค่าเป็นชื่อของเดือน (เช่น September) จากแป้นพิมพ์ และทำการตรวจสอบว่าผู้ใช้พิมพ์ชื่อเดือนถูกต้องหรือไม่ หากพิมพ์ถูกต้องให้พิมพ์ผลลัพธ์เป็นตัวอักษร 3 ตัวแรก เช่น ถ้า month = “September” ผลลัพธ คือ “Sep” หากพิมพ์ไม่ถูกไม่ถูกต้องให้แสดง Error ออกหน้าจอโดยมีการรับและแสดงผลข้อมูลดังตัวอย่าง

รายชื่อเดือนทั้ง 12 เดือน

"January","February","March","April","May","June","July","August","September","October","November","December
“

รายชื่อที่มาของชื่อเดือนทั้ง 12 เดือน

"Janus","Februus","Martius","Aperire","Maia","Juno","Julius","Caesar","Augustus","Caesar","Septem","Octo","Novem","Decem"

ตัวอย่างการรับและแสดงผลข้อมูล

```
Enter month : March
Month : Mar
```

```
Enter month : Jan
Error Incorrect data
```

Assignment 5 : Python program for encoding message

จงเขียนโปรแกรมเข้ารหัสข้อมูล กำหนดให้รับข้อความ และตัวอักษรที่ต้องการให้เข้ารหัส และค่ารหัสทางแป้นพิมพ์ โดยมีการรับและแสดงผลข้อมูลดังตัวอย่าง

ตัวอย่างการรับและแสดงผลข้อมูล

```
Enter String : I love python program
Enter Characters : o
Enter Encoded : #
Encrypt string =  I l#ve pyth#n pr#gram
```

Assignment 6 : Python program for encoding message

จงเขียนโปรแกรมรับข้อความมา 1เพื่อทำการเข้ารหัสข้อความสำหรับป้องกันความปลอดภัย (Encoding) โดยหากเป็นสระให้เพิ่มค่าไปอีก 2 ตำแหน่งนอกนั้นเพิ่ม 1 ตำแหน่ง แสดงผลออกหน้าจอดังตัวอย่าง

ตัวอย่างการรับและแสดงผลข้อมูล

```
Input text : Hello my name is pojanee juntarasupawong
-----encoding----- by replace vowel in order+2
Igmmq!nz!ocng!kt!qqkcogg!kwoucsctwqcxqoh
```