Working with Text

text plays a central role in computer programs.

- email readers
- web browsers
- calendars
- strand of DNA
- games, ….

we will show how to make programs to be more interactive by printing messages to our programs' users and getting input from them.

Creating Strings of Characters

What is string?

String any sequence of characters between quotation mark, double, single or triple quotation.

```
>>> “ this is a string”
>>> s =   “ this is a string”
>>> type(s)
```

Operations on Strings

There are so many operation on string we will explain some of them in this lecture

You can find these operation in the following link:
https://docs.python.org/2/library/string.html

Len: returns the number of characters between the opening and closing quotes:

>>> len('Albert Einstein')
15
>>> len('123!')
4
>>> len(' ')
1
>>> len('') ??

String Concatenation

We can add two strings using the + operator, which produces a new string containing the same characters as in the two operands:

>>> 'Albert' + ' Einstein‘

'Albert Einstein'

When + has two string operands, it is referred to as the *concatenation operator*.

WARNING: Don’t copy string with quote from MS office Application , it has different ASCCII representation

String Concatenation -cont

Adding an empty string to another string
produces a new string that is just like the nonempty operand:

```
>>> "Alan Turing" + ''
'Alan Turing'
>>> "" + 'Grace Hopper'
'Grace Hopper'
```

Can operator + be applied to a string and a numeric value? If so, would addition or concatenation occur? give it a try:

>>> 'NH' + 3

>>> 9 + ' planets'

Traceback (most recent call last):

File "<stdin>", line 1, in <module>

TypeError: unsupported operand type(s) for +: 'int' and 'str‘

Here, because Python saw a 9 first, it expected the second operand to also be numeric.

How to add number to string

join a string with a number, function ***str*** can be applied to the number to get a string representation of it, and then the concatenation can be done:

>>> 'Four score and ' + str(7) + ' years ago‘

'Four score and 7 years ago'

Function ***int*** can be applied to a string whose contents look like an integer,
and float can be applied to a string whose contents are numeric:

>>> int('0')
0
>>> int("11")
11
>>> int('-324')
-324
>>> float('-324')
-324.0
>>> float("56.34")
56.34
BUT Can we get:
>>> int('a') **or** Float(‘b’)

operator *

A string can be repeated using operator * and an integer, like this:

>>> 'AT' * 5
'ATATATATAT‘

>>> 4 * '-'
'----‘

If the integer is less than or equal to zero, the operator yields the empty string (a string containing no characters):
>>> 'GC' * 0
''
>>> 'TATATATA' * -3 ????

String is a value

Strings are values, so you can assign a string to a variable. Also, operations on strings can be applied to those variables:

```
>>> sequence = 'ATTGTCCCCC'
>>> len(sequence)
10

>>> new_sequence = sequence + 'GGCCTCCTGC'
>>> new_sequence
'ATTGTCCCCCGGCCTCCTGC‘

>>> new_sequence * 2
'ATTGTCCCCCGGCCTCCTGCATTGTCCCCCGGCCTCCTGC'
```

Using Special Characters in Strings

Mixing of quotations:

The opening and closing quotes of a string must match:

DO you think the following string will work? Why

>>> 'Charles Darwin"

will the following string work? Why

>>> ‘Let’s play tennis’

When Python encounters the second quote—the one that is intended to be part of the string—it thinks the string is ended. Then it doesn’t know what to do with the text that comes after the second quote.

Using Special Characters in Strings

How about this string

>>>”let’s play tennis”

How about this string

```
>>> s='Einstein said:"Things should be as simple as possible but not
simplear" '
```

Using Special Characters in Strings

Other way if special character has meaning in the interpreter and you want to ignore then you can use back slash \ then the effect will be ignored.

will the following string work? Why

>>> 'Let's play tennis'

will the following string work? Why

>>> 'Let\'s play tennis'

What is the result of this function

```
>>> len('it\'s')
```

Escape Sequences in string

Escape Sequence	Description
\'	Single quote
\"	Double quote
\\	Backslash
\t	Tab
\n	Newline
\r	Carriage return

What is the difference between \n and \r

Creating a Multiline String

creating a string using single or double quotes, the whole string must fit onto a single line.

Ex: >>>'the weather today is very nice'

Here's what happens when you try to stretch a string across multiple lines:

>>> '''one

... two

... three'''

Still you get one line .. So how to get multiple line?

Like that?

>>>'one\ntwo\nthree' **Still you get one line**

We have to use print function:

>>>print('one\ntwo\nthree')

how to separate 'one two three' with tab? Do it!!

Exercise:
create the following string output in one python statement :

one two
three four

To sum up: \n is working only in function print
EX:

```
>>> numbers = '''one
... two
... three'''
>>> numbers                    ?

>>> print(numbers)             ?
```

(Try it)

You can combine two or more string

```
>>> s="rock"
>>> t="climbing"
>>> s+t
'rockclimbing'
>>> s+" "+t
'rock climbing'
```

Function print can print values of any type, and it can even print values of different types in the same function call:

>>> print(1, 'two', 'three', 4.0)

Also we can formulate via print function the format of our output:

```
>>> radius = 5
>>> print("The diameter of the circle is", radius * 2, "cm.")
```

The diameter of the circle is 10 cm.

To can get help via:

Help(print)

Space and separation customization using print

```
>>> print('a', 'b', 'c') # The separator is a space by default
a b c
>>> print('a', 'b', 'c', sep=',      ')
```

Try it

You can tell Python IDLE to end with an empty string instead of a new line by assigning the prompt sign >>> at the end of the string:

```
>>> print('a', 'b', 'c', sep=', ', end='')
```

Try it

Getting Information from the Keyboard

So far all input to the python is given in the same code
How can we want to create conversational mode of the python?

This could be done via : input statement:

```
>>> species = input()
Homo sapiens
>>> species
'Homo sapiens'
>>> population = input()
6973738433
>>> population
'6973738433'
>>> type(population)
<class 'str'>
```

We can not make calculation on population variable

Getting Information from the Keyboard

We can input any variable type by assigning the type in front of the input

```
>>> population = input()
6973738433
>>> population
'6973738433‘
>>>type(population)
>>> population = int(population)
>>> population
6973738433
>>>type(population)
>>> population = population + 1
>>> population
6973738434
```

Getting Information from the Keyboard

If you want make friendly input

>>> population = input("Please give the population : ")
Please give the population : 345676

Getting help string:

```
import math
def area_of_circle(radius):
    '''
    (number, int)->number
    Returns the area of the circle of the given radius

    Precondtion: radious is non negative number
    '''
    return math.pi*radius**2
```

Summery:

In this chapter, you learned the following:

- ✓ Python uses type str to represent text as sequences of characters.
- ✓ Strings are created by placing pairs of single or double quotes around the text.
- ✓ Multiline strings can be created using matching pairs of triple quotes.
- ✓ Special characters like newline and tab are represented using escape sequences that begin with a backslash.
- ✓ Values can be printed using built-in function print, and input can be provided by the user using built-in function input.