



香港中文大學(深圳)

The Chinese University of Hong Kong, Shenzhen

# ***INTRODUCTION TO COMPUTER SCIENCE: PROGRAMMING METHODOLOGY***

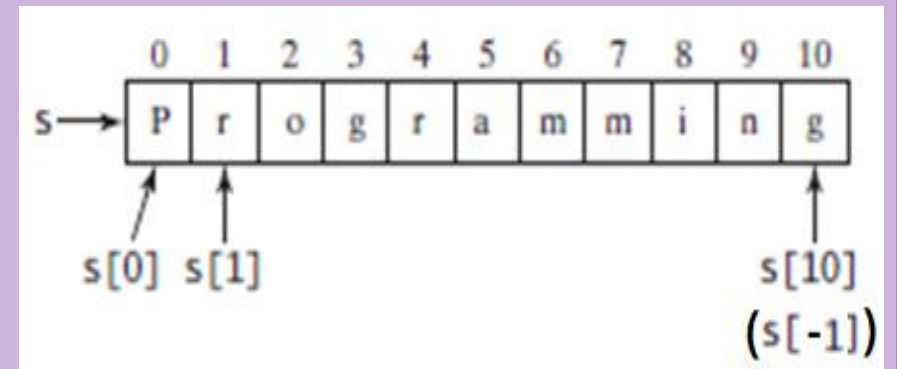
## TUTORIAL 6

### STRINGS AND FILES

# String-Index operator

索引操作符

➤ 1. Index operator `[]`: used to visit the elements in a string `s`, `s[i]`,  $i = 0, 1, 2, \dots, \text{len}(s) - 1$  represents the  $(i+1)$ th element, or `s[i]`,  $i = -1, -2, \dots, -\text{len}(s)$  represents the  $(\text{len}(s)+i+1)$ th element. Notice: string's indexes start from 0.



➤ 2. Slicing operator `[start:end]` ( $\text{start} \rightarrow \text{end}-1$ ): `s[start:end]` cuts the string `s` beginning from elements with index `start` to `end-1`.

左闭右开

# Manipulating strings

- 3. A way to delete an element in a string `s`: `s=s[:i]+s[i+1:]` can delete the element with index `i`.
- 4. A way to change an element in a string `s`: example:  
`s= 'Walcome' , then s=s[:1]+' e' +s[2:]` will change `s` to 'Welcome'. Notice: Strings' elements cannot be directly changed. So here it is **wrong** to write `s[1]= 'e' .`
- 5. `len(s)` returns the length of string `s`.
- 6. “in” operator: character `ch` is included in string `s`, **`ch in s`**→True, otherwise **`ch in s`**→False.
- 7. Strings can also act as a collection of elements in for loop:  
**`for i in s:`**

# Some methods of strings-I

- 8. Methods in str class to judge types of characters in strings.
  - i) `s.isalnum()`, ii) `s.isalpha()`, iii) `s.isdigit()`, iv) `s.islower()`, v) `s.isupper()` judge whether s contains i) only letters and numbers, ii) only letters, iii) only numbers, iv) all letters in s are lowercase, v) all letters in s are uppercase. **True or False** will be returned.
- 9. Methods in str class to manipulate substrings.

子字符串s1

  - i) `s.endswith(s1)` returns True if the string s ends with substring s1.
  - ii) `s.startswith(s1)` returns True if the string s starts with substring s1.
  - iii) `s.find(s1)` returns the lowest index where s1 starts in the string s, or -1 if s1 is not found; similarly, `s.rfind(s1)` returns the highest index.
  - iv) `s.count(s1)` returns the number of non-overlapping occurrences of the substring s1.

计算字符串s中子字符串  
s1的出现次数

# Some methods of strings-II

- 10. Methods in str class to convert strings.
  - i) `s.lower()`: returns a copy of the string `s` with all letters converted to lowercase. Similarly, `s.upper()` uppercase.
  - ii) `s.swapcase()`: returns a copy of the string `s` in which lowercase letters are converted to uppercase and uppercase to lowercase.
  - iii) `s.replace(old,new)`: returns a new string that replaces all the occurrences of the old substring in `s` with a new substring.
- 11. Splitting a string into a list: examples:
  - i) `items= 'Jane John Peter Susan' .split()` -  
→ `[ 'Jane' , ' John' , ' Peter' , 'Susan' ]`
  - ii) `items=' 03/07/2019' .split( '/' )`→ `[ '03' , ' 07' , ' 2019' ]`.

将字符串中的某个子字符串替换为另一个子字符串

# Text input and output-I

`fileVariable = open(filename, mode)`

➤ **Three modes:**

**“r” – Opens a file for reading**

**“w” – Opens a file for writing**

**“a” – Opens a file for appending data from the end of the file**

`read([number.int]): str`

`readline(): str`

`readlines(): list`

`write(s: str): None`

`close(): None`

Returns the specified number of characters from the file. If the argument is omitted, the entire remaining contents in the file are read.

Returns the next line of the file as a string.

Returns a list of the remaining lines in the file.

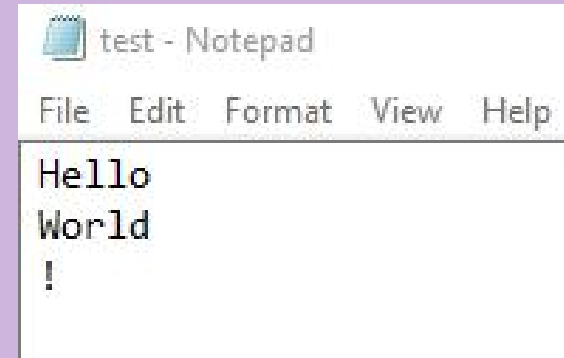
Writes the string to the file.

Closes the file.

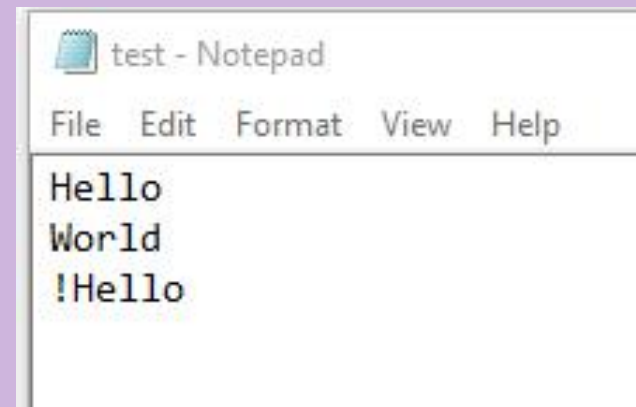
# Text input and output-II

- Write function will write strings to a file. If it does not exist, a new file will be created.

```
file=open("test.txt",'w')  
file.write('Hello\nWorld\n!')  
file.close()
```



```
file=open("test.txt",'a')  
file.write('Hello')  
file.close()
```



# Text input and output-III

```
file=open("test.txt",'r')
print(file.read())
file.close()
```

```
Hello
World
!Hello
>>>
```

➤ Return all the content.

```
file=open("test.txt",'r')
print(file.read(3))
file.close()
```

```
Hel
>>>
```

➤ Return 3 characters.

```
file=open("test.txt",'r')
print(file.readline())
print(file.readline())
file.close()
```

```
Hello
World
>>>
```

➤ Return a line.

```
file=open("test.txt",'r')
print(file.readlines())
file.close()
```

```
['Hello\n', 'World\n', '!Hello']
>>>
```

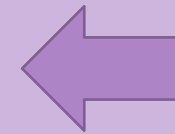
➤ Return every line as a list.



# Q1: Check substrings

- You can check whether a string is a substring of another string by using the find method in the str class. Write your own function to implement find. Write a program that prompts the user to enter two strings and then checks whether the first string is a substring of the second string.

```
first = input('Please enter the first string: ')
second = input('Please enter the second string: ')
if second.find(first) != -1:
    print('first is a sbustring of second')
else:
    print('first is not a sbustring of second')
```



Write your own  
program to  
realize the  
function of  
find method.

## Q2: Check password

➤ Some Web sites impose certain rules for passwords. Write a function that checks whether a string is a valid password. Suppose the password rules are as follows:

- A password must have at least eight characters. `len()`
- A password must consist of only letters and digits. `isalnum()`
- A password must contain at least two digits. `isdigit()`

## Q3: Longest common prefix

- Write a method that returns the longest common prefix of two strings. For example, the longest common prefix of **distance** and **disinfection** is **dis**. The header of the method is **def prefix(s1,s2):** . If the two strings have no common prefix, the method returns an empty string. Write a main method that prompts the user to enter two strings and display their longest common prefix.

```
Enter the first string:distance
Enter the second string:disinfection
The longest common prefix of the two strings is "dis".
```

## Q4: Handling files

- i) Write a program that will count the number of characters, words, and lines in a file. Words are separated by a whitespace character. Your program should prompt the user to enter a filename.

```
Enter a filename:JaneEyre.txt
In the file JaneEyre.txt, there are
10807 characters,
1930 words,
46 lines.
```

- ii) In the file “JaneEyre.txt”, there are some typos “amd” which should actually be “and”, find how many of them and replace all, then write the correct version into a new file.