香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

INTRODUCTION TO COMPUTER SCIENCE: PROGRAMMING METHODOLOGY

TUTORIAL 3

PYTHON BASICS

CHEN WANYI

Outline

- 1. Data types, e.g. integer, string, boolean etc.
- 2. print() function.
- 3. Some arithmetic operators, e.g. +, -, *, /, //, %, ** etc.
- 4. input() function.
- 5. eval() function.

I .Data Types

- There are several data types very often used:
  i)**Integer**, e.g. 1,19,-36...
  ii)**Floating-point number**, e.g. 3.14159, 9.80...
  iii)**String**, e.g. 'hello world', '123', 'www.cuhk.edu.cn'...
  iv)**Boolean**, the value can only be True or False, e.g. 5>9, 'b'<'a',1=='1'...
  v)**List**, e.g. [1,2,3], ['h', 'e', 'l', 'l', 'o']...
- Assign some value of different types to a variable will make it become different data types.
- Using type() function you can check the data type of a variable.
- Data types can be forced to change if possible,
  e.g. int(9.8)->9, str(123)->'123', float('9.8')->9.8,
  list('hello')-> ['h', 'e', 'l', 'l', 'o'] ...

2.print() Function-I

- print() is a function-something you can realize some functionalities by calling it.
- The "()" is for you to put in some variables as input.
- You can use help(print) in shell to check how it is used.
- The parameters usually used in print() function is print(value, sep, end).
  - "value" is the content you want to print out, e.g. print("hello"), print(1,2,3) etc.
  - "sep" is the symbol you use to separate many values, by default it is ' ' if without mentioned. But you can change it to ',', '-'or '*' ... e.g. print(1,2,3,sep=',').
  - "end" is the symbol you use to end you output, by default it is '\n'if not being mentioned. You can change it to ' ', ';' or 'end'... e.g. print('hello', end='#').

```
>>> a=1
>>> b=2
>>> c=3
>>> print(a, b, c)
1 2 3
>>> print(a, b, c, sep=';')
1;2;3
>>> print(a, b, c, end=';')
1 2 3;
```

2.print() Function II-Formatted Output-A

```
>>> a=1.0
>>> print('%d'%a)
1
>>> print('%f'%a)
1.000000
>>> print('%s'%a)
1.0
```

- "d" in print('%d'%v) means the variable v is integer type;
  "f" in print('%f'%v) means the variable v is a floating point number;
  "s" in print('%s'%v) means the variable v is a string.

```
>>> a=1
>>> print('%8d'%a)
       1
>>> print('%7.2f'%a)
   1.00
```

- Determine the space for the variable to be shown:
  the number('8') before 'd' in print('%8d'%v) means 8 spaces for the integer v;
  the number('7.2') before 'f' in print('%7.2f'%v) means 7 spaces for the floating point number v while keeping the 2 digits after decimal point '.'.
- To align the result leftward, add the symbol '-': print('%-8d'%v) will make the integer v printed from the left of the 8 spaces. Without '-' will align the result rightward.

```
>>> a=1
>>> b=2
>>> print('%-8d\n%8d'%(a,b))
1
       2
```

2.print() Function III-Formatted Output-B

- To print many values within some predefined formats:

  print('%s has %d petals and is worth %.2f yuan a bunch.'%(FlowerName, NumOfPetal, Price))

  where the variable 'FlowerName' is the name of a flower(string type),
  'NumOfPetal' is the number of petals of the flower(integer type),
  'Price' is the price of the flower (floating point number).

```
>>> a=1
>>> b=2
>>> print('a+b')
a+b
>>> print(a+b)
3
>>> print('a is %d; b is %d; a+b=%d'%(a,b,(a+b)))
a is 1; b is 2; a+b=3
>>> print('a is',a,'; b is',b,'; a+b=%d'%(a+b))
a is 1 ; b is 2 ; a+b=3
```

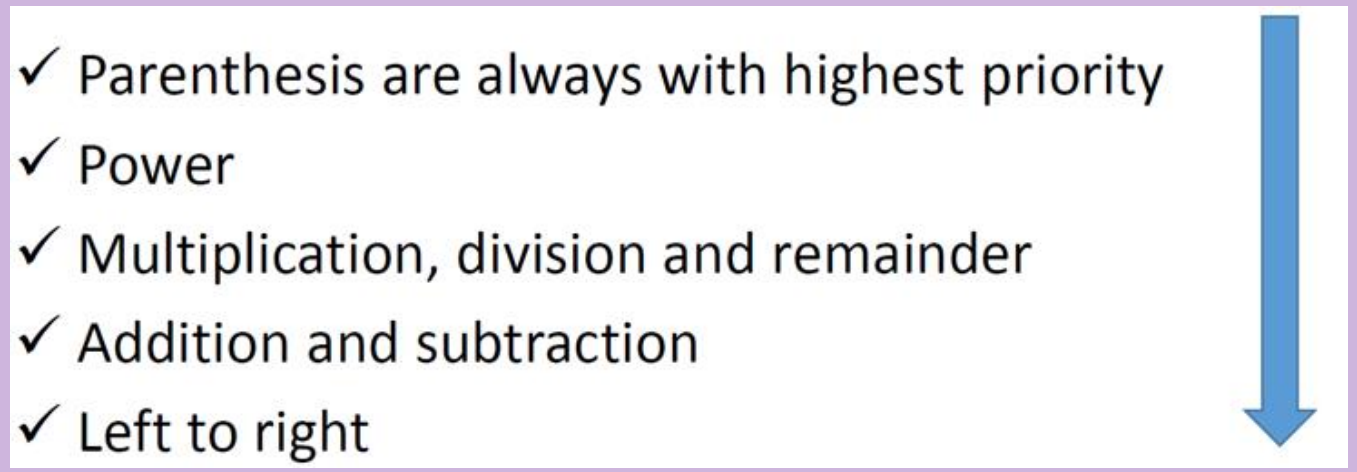
Practice 1:Print a table

- Write a program that displace the following table(8 spaces for each numbers and they are aligned leftward):

a	b	a ** b
1	2	1
2	3	8
3	4	81
4	5	1024
5	6	15625

3.Arithmetic Operators

- Some arithmetic operators often used: Addition +, Subtraction -, Multiplication *, Division /, Modulus(Remainder) %, Exponent(Power) **, Floor division // etc.
- Order of operations(Operator precedence):



- Augmented assignment operators: “x+=1” means “x=x+1”, “x//=10” means “x=x//10”, similar for “x-=1”, “x%=10”, “x**=2”,...

3.Arithmetic Operators

- Some operators for string type variables:

+: Concatenation - Adds values on either side of the operator

*: Repetition - Creates new strings, concatenating multiple copies of the same string

[ ]: Slice - Gives the character from the given index

[:]: Range Slice - Gives the characters from the given range

```
>>> a='test1'
>>> b='test2'
>>> a+b
'test1test2'
>>> a*3
'test1test1test1'
>>> a[0]
't'
>>> a[0:3]
'tes'
>>> a*b
Traceback (most recent call last):
  File "<pyshell#168>", line 1, in <module>
    a*b
TypeError: can't multiply sequence by non-int of type 'str'
>>> a**3
Traceback (most recent call last):
  File "<pyshell#169>", line 1, in <module>
    a**3
TypeError: unsupported operand type(s) for ** or pow(): 'str' and 'int'
```

4.input() Function

- input() is a function used for instructing the users to input a string for a variable. Notice that the data type that input() function return is a string, that is, if you assign the result to a variable e.g. v=input(), then v is a string data type.
- You can use a string as a variable for the input() function as an instruction, e.g. input("Please input a number:")

```
>>> a=input('Please enter a number: ')
Please enter a number: 2
>>> a
'2'
>>> type(a)
<class 'str'>
```

5.eval() Function

- eval() is a function used to parse and evaluate the Python expression(a string) you put as a variable for the eval() function, i.e. as if the '' of the expression is taken away.

- eval() is often used together with input() in order to change the string, which is returned by input() function, into the data type you want, e.g. N=eval(input()) will make the value passed to variable N be an integer, a floating-point number or a boolean if you input 9, 3.14 or 3>5 respectively, for instance.

```
>>> a=input('Please enter a number: ')
Please enter a number: 1
>>> type(a)
<class 'str'>
>>> b=eval(input('Please enter a number: '))
Please enter a number: 1
>>> type(b)
<class 'int'>
```

```
>>> input('Please enter a number:')
Please enter a number:test
'test'
>>> eval(input('Please enter a number: '))
Please enter a number: test
Traceback (most recent call last):
  File "<pyshell#199>", line 1, in <module>
    eval(input('Please enter a number: '))
  File "<string>", line 1, in <module>
NameError: name 'test' is not defined
```

5.eval() Function

- Example: what are the outputs of the following cases?

i)

```
a=1
b=3
c='a+b'
d=eval('a+b')
print(c)
print(d)
```

ii)

```
a=2
aa=4
aaa=8
b=eval('a+aa')
print(b)
```

Practice 2:Convert Celsius to Fahrenheit

- Write a program that reads a Celsius degree from the console and converts it to Fahrenheit and displays the result. The formula for the conversion is as follows:

$$\mathbf{fahrenheit} = (9/5) * \mathbf{celsius} + 32$$

Here is a sample:

```
Enter a degree in Celsius:43
Celsius degree 43 equals to Fahrenheit degree 109.4.
```

Practice 3:Compute the volume

- Write a program that reads in the radius and length of a cylinder and computes the area and volume using the following formulas:

$$\mathbf{area} = \mathbf{radius} * \mathbf{radius} * \boldsymbol{\pi}$$

$$\mathbf{volume} = \mathbf{area} * \mathbf{length}$$

Here is a sample:

```
Enter radius of the cylinder:12
Enter length of the cylinder:33
The area is:452.39
The volume is:14928.85
```

Practice 4: Sum the digits

- Write a program that reads an integer between 0 and 1000 and adds all the digits in the integer. For example, if an integer is 932, the sum of all its digits is 14. (Hint: Use the % operator to extract digits, and use the // operator to remove the extracted digit. For instance, 932%10=2 and 932//10=93.)

Here is a sample:

```
Enter a number between 0 and 1000:932
Sum up all the digits as: 14
```

Practice 5: Number of years and days

- Write a program that prompts the user to enter the minutes (e.g., 1 billion), and displays the number of years and days for the minutes. For simplicity, assume a year has 365 days.

Here is a sample:

```
Enter the number of minutes:1000000000
That minutes will equal to 1902 years and 214 days.
```

Practice 6: Financial application: compound value

- Suppose you save $100 each month into a saving account with monthly interest rate 0.417%. That is, i)after the first month, the value in the account becomes 100*(1+0.00417)=100.417; ii) after the second month, the value in the account becomes (100+100.417)*(1+0.00417) =201.252; iii)after the third month, the value in the account becomes (100+ 201.252)*(1+0.00417)=302.507; and so on. Write a program that prompts the user to enter a monthly saving amount and displays the account value after the sixth month.

Here is a sample:

```
Enter the monthly saving amount:$100
After the sixth month, the account value is:$608.82
```