Ch3: Python Basic Part II

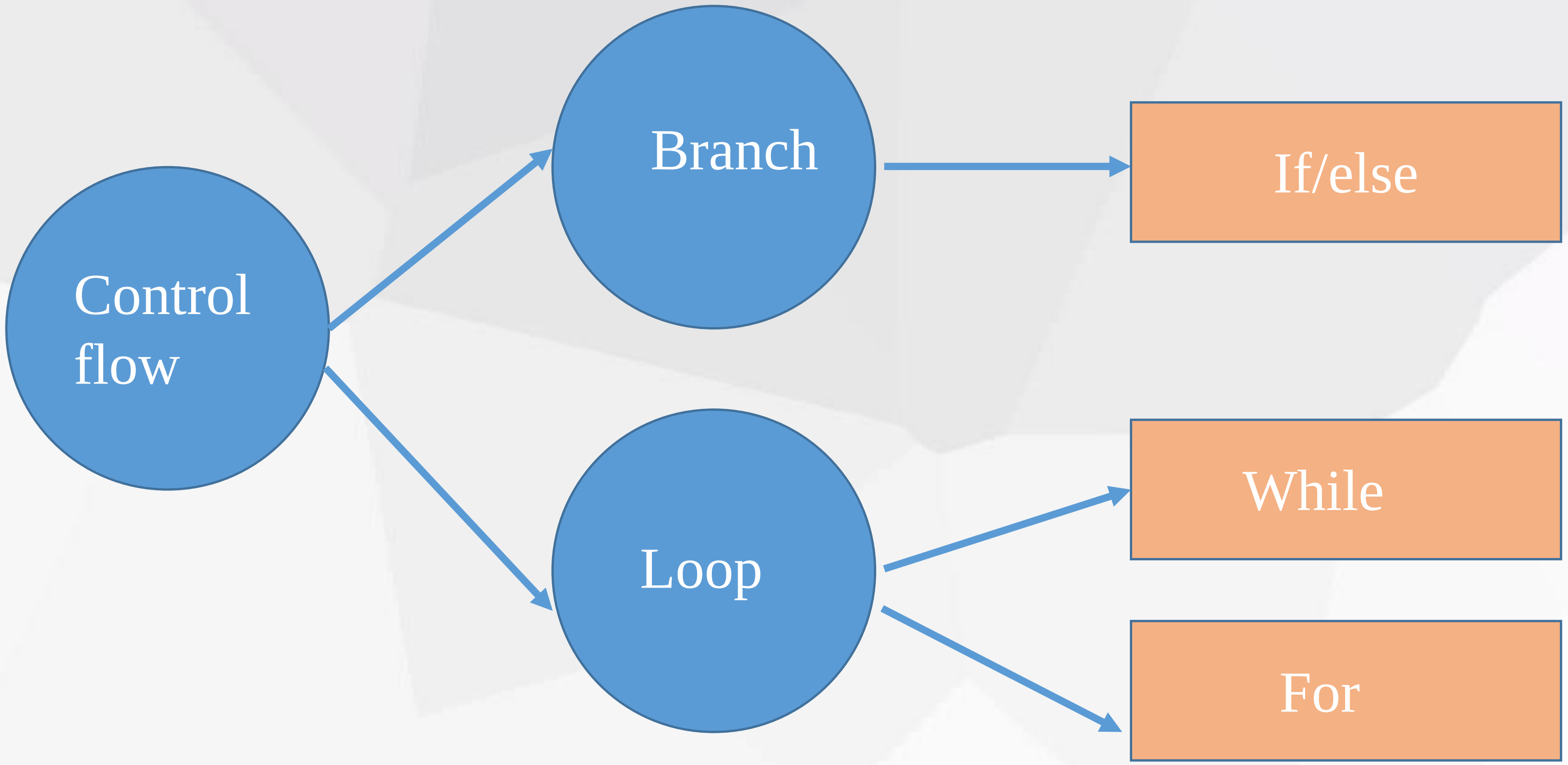
Programming with Python

SOUTH CHINA UNIVERSITY OF TECHNOLOGY

DR. MAO AIHUA
AHMAO@SCUT.EDU.CN

Control Flow

Knowledge Graph



单选题 1分

设置

Regarding the branch structure of Python, which one is wrong?

A The if-elif-else statement in Python describes the multi-branch structure

[✓ B] The branch structure can jump back to the statement that has been executed

C if can be used in branch structure

D The if-else statement in Python is used to form a two-branch structure

提交

单选题 1分

The output of the following program is: ()

```
 t = "Python"
if t>="python":
   t = "python"
else:
   t = "None"
print(t)
```

A t

B Python

[✓ C] None

D python

An Example of Educational Game

- Choose a direction: left, right, up, down
- Draw a number for steps

Control flow

Life Is Full of Choice

Control flow

- Life is full of choices and a learning experience.
- Everything that happens to us in life is a result of our choices.
- The choices you make today will impact your life tomorrow.
- Respect **your and also other’s choice**.

Control Flow - Branching

```
if <condition>:
    statements
```

```
if <condition>:
    statements

else:
    statements
```

```
if <condition>:
    statements

elif <condition>:
    statements
else:
    statements
```

- <condition>has a **Boolean** value: Trueor False
- relies on indentation to define scope in the code

Control Flow-If …

- The if loop has the following general form.

```
if <condition>:
    statements
```

- If condition evaluates to **True**, the statements are executed. Otherwise, they are skipped entirely.

Control Flow-If ...

```
a = 1
b = 0
if a:
    print("a is true!“)
if not b:
    print("b is false!“)
if a and b:
    print("a and b are true!“)
if a or b:
    print("a or b is true!" )
```

What is the output?

```
a is true!
b is false!
a or b is true!
```

Control Flow - Branching

- Boolean conditions
  - Equals: a == b
  - Not Equals: a != b
  - Less than: a < b
  - Less than or equal to: a <= b
  - Greater than: a > b
  - Greater than or equal to: a >= b

- We can also pair an else with an if branch.

```
if condition :
        statements

else:
        statements
```

```
a = 1
b = 2
c = 2
if a > b:
    print("a is greatest")
else:
    print("b is greatest")
```

What is the output?

```
b is greatest
```

- The **elif** keyword can be used to specify an else if branch.

```
if condition :
        statements
else if condition:
         statements
else:
        statements
```

```
a = 0
b = 2
c = 1
if a > b:
   print("a is greatest")
elif b > c:
   print("b is greatest")
else:
   print("c is greatest")
```

What is the output?

```
b is greatest
```

- Furthermore, if statements may be nested within each other.

```
if condition :
        if condition:
                statements
        else:
                statements
else:
        statements
```

```
a = 1
b = 0
c = 2
if a > b:
    if a > c:
        print("a is greatest")
    else:
        print("c is greatest")
elif b > c:
    print("b is greatest")
else:
    print("c is greatest")
```

What is the output?

```
c is greatest
```

```
num = 5
if num == 3:
    print('boss')
elif num == 2:
    print('user')
elif num == 1:
    print('worker')
else:
    print('roadman' )
```

What is the output?

```
var = 100
if var < 200:
    print("Expression value is less than 200")
    if var == 150:
        print("Which is 150")
    elif var == 100:
        print("Which is 100")
    elif var == 50:
        print("Which is 50")
elif var < 50:
     print("Expression value is less than 50")
else:
     print("Could not find true expression")
```

What is the output?

Control Flow- Loop

- **Python has two primitive loop commands**
  - ➢ **while** loops :

    we can execute a set of statements as long as a condition is true
  - ➢ **for** loops :

    used for iterating over a set of statements with a fixed number of times

单选题 1分

设置

What will be the output of the following Python code?

```
x = 'abcd'
for i in x:
        print(i)
```

A a B C D

[✓ B] a b c d

C A B C D

D error

提交

Control Flow- Loop

- **While** loops (repeat implementing) have the following general structure.

```
while condition:
        statements
```

- Here, *statements* refers to one or more lines of Python code, and considered as a block of code
- The *condition* may be any expression, where **any non-zero value is true**. The loop iterates while the expression is true.

```
count = 0
while (count < 3):
    count = count + 1
    print("Hello Python")
```

What is the output?

```
Hello Python
Hello Python
Hello Python
```

Control Flow- Loop

```
i = 1
while i < 4:
    print(i)
    i = i + 1
flag = True
while flag and i < 8:
    print(flag, i)
    i = i + 1
```

What is the output?

```
1
2
3
True 4
True 5
True 6
True 7
```

Control Flow- Loop

```
i = 0
result = 0
while i<= 10:
    result += i
    i += 1
print(result)
```

What is the output?

```
55
```

Control Flow- Loop

- **For** loop has the following general form.

```
for var in sequence:
    statements
```

- Sequence is a collection of sequence objects like list, tuple
- Each item in the sequence is assigned to *var*, and the statements are executed until the entire sequence is exhausted.

```
for letter in "aeiou":
    print("letter: ", letter)

for i in [1,2,3]:
    print(i)
```

What is the output?

```
letter: a
letter : e
letter : i
letter: o
letter : u
1
2
3
```

Control Flow- Loop

- **For** loop has the following general form.

```
for var in sequence:
    statements
```

- Sequence is a collection of sequence objects like list, tuple
- Each item in the sequence is assigned to *var*, and the statements are executed until the entire sequence is exhausted.

```
# Iterating over a list
l = ["I", "love", "python"]
for i in l:
    print(i)
```

What is the output?

```
I
love
python
```

```
# Iterating over a tuple
t = ("It", "is", "fine")
for i in t:
    print(i)
```

What is the output?

```
It
is
fine
```

Control Flow- Loop

- For loops may **be nested** with other control flow tools such as while loops and if
-

```
for letter in "aeiou":
    if letter!='e':
        print("letter: ", letter)
```

What is the output?

```
letter:  a
letter:  i
letter:  o
letter:  u
```

Control Flow- Loop

- For loops may even **be nested** with another for statements.

```
for letter in "aeiou":
    for i in (0,1):
        print("letter: ", letter,i)
```

What is the output?

```
letter:  a 0
letter:  a 1
letter:  e 0
letter:  e 1
letter:  i 0
letter:  i 1
letter:  o 0
letter:  o 1
letter:  u 0
letter:  u 1
```

主观题 10分

设置

- For loops may even **be nested** with another for statements.

```
for x in "12ab":
    print("Hello World", x)
```

```
for x in "ABC":
  for y in "123":
     print(x+y)
```

What is the output?

正常使用主观题需2.0以上版本雨课堂

作答

Control Flow- Loop

- In Python, **range()** is a handy built-in functions for creating a range of integers, typically used in for loops.

```
for i in range(0,3):
    print(i)
```

- Here range(0,3) generate the integer sequence of 0,1,2

What is the output?

```
0
1
2
```

Control Flow Tools

- **Syntax of range()**

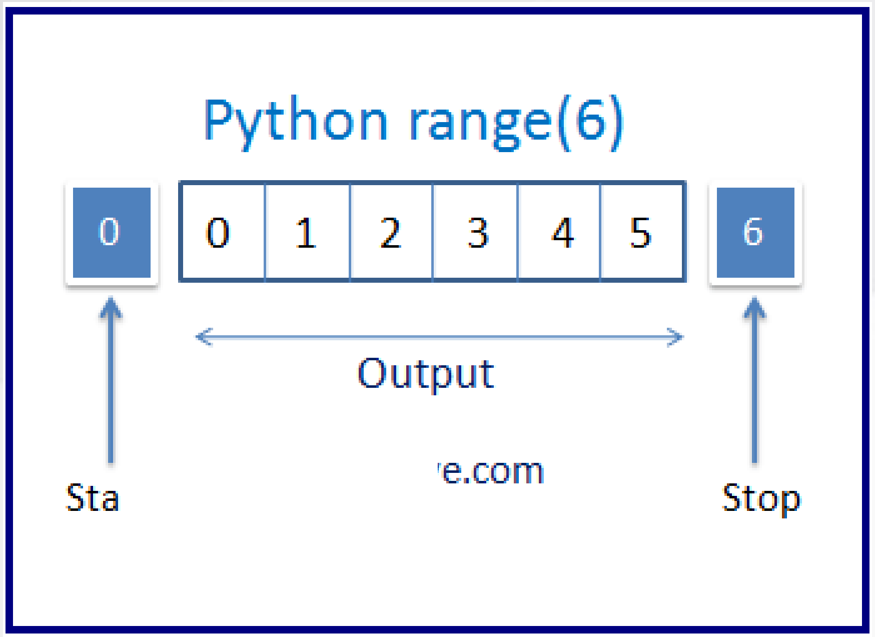
  range *(start*, *stop*, *step*)

Parameter	Description
start	Optional. An integer number specifying at which position to start. Default is 0
stop	Required. An integer number specifying at which position to stop (not included).
step	Optional. An integer number specifying the incrementation. Default is 1

Control Flow Tools

- **range(6)**

We got integers from 0 to 5 because range() function doesn't include the last (stop) number in the result.



0, 1, 2, 3, 4, 5

Control Flow Tools

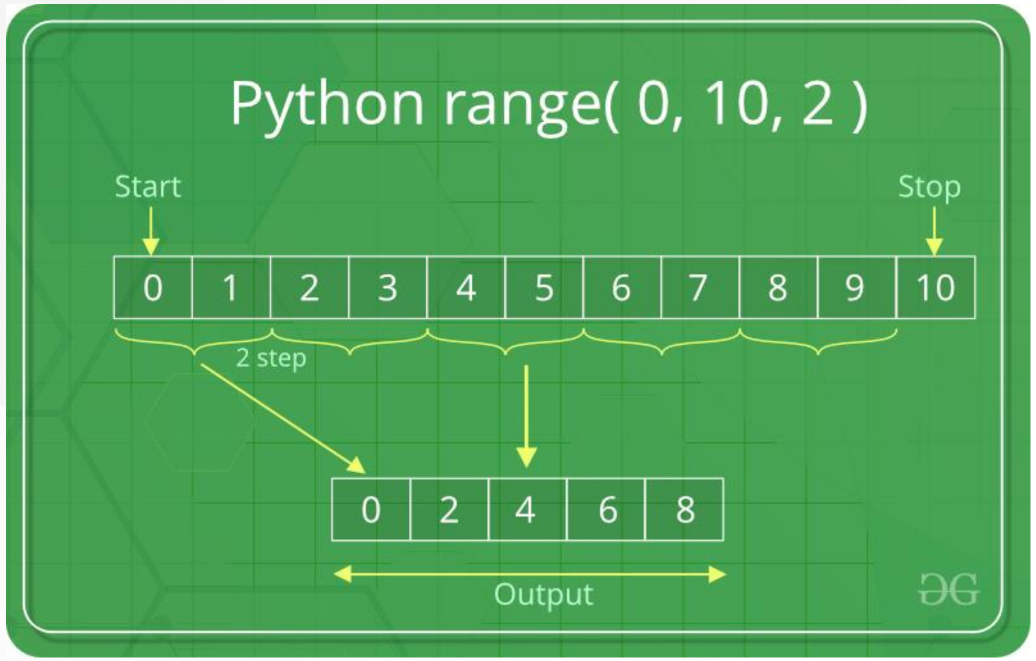
- **range(5,10)**

Here, start is set as 5, we got integers from 5 to 9

5, 6, 7, 8, 9

- **range(0,10, 2)**

Here, start is set as 0, and step is set as 2, we got integers 0,2,4,6,8



主观题 10分

设置

```
for i in range(0,3):
   for j in range(0,3):
         print(i+j)
```

What is the output?

```
for i in range(0, 4):
    print(i)
for i in range(0,8,2):
    print(i)
for i in range(20,14,-2):
    print(i)
```

What is the output?

作答

Control Flow Tools

- There are statements provided for manipulating loop structures.
  - ➢ **break, continue, pass**
- **Break**: terminates the current loop.
- **Continue**: immediately begin the next iteration of the loop, and the current iteration of the loop will be disrupted.
- **Pass:** do nothing. Use when a statement is required syntactically.

Break Statement

```
number = 0
for number in range(10):
    if number == 5:
        break    # break here
    print('Number is ' + str(number))
print('Out of loop')
```

- What are results in this program?

```
Number is 0
Number is 1
Number is 2
Number is 3
Number is 4
Out of loop
```

Continue Statement

```
number = 0
for number in range(10):
    if number == 5:
        continue    # continue here
    print('Number is ' + str(number))
print('Out of loop')
```

- what happens in this program?

```
Number is 0
Number is 1
Number is 2
Number is 3
Number is 4
Number is 6
Number is 7
Number is 8
Number is 9
Out of loop
```

Pass Statement

```
number = 0
for number in range(10):
    if number == 5:
        pass    # pass here
    print('Number is ' + str(number))
print('Out of loop')
```

- what happens in this program?

```
Number is 0
Number is 1
Number is 2
Number is 3
Number is 4
Number is 5
Number is 6
Number is 7
Number is 8
Number is 9
Out of loop
```

主观题 10分

设置

```
for j in range(10):
   if j > 5 and j <= 8:
       continue
       print("continue case")
   print(j)
```

```
for j in range(10):
   if j > 5 and j <= 8:
       print("continue case")
       break
   print(j)
```

正常使用主观题需2.0以上版本雨课堂

作答

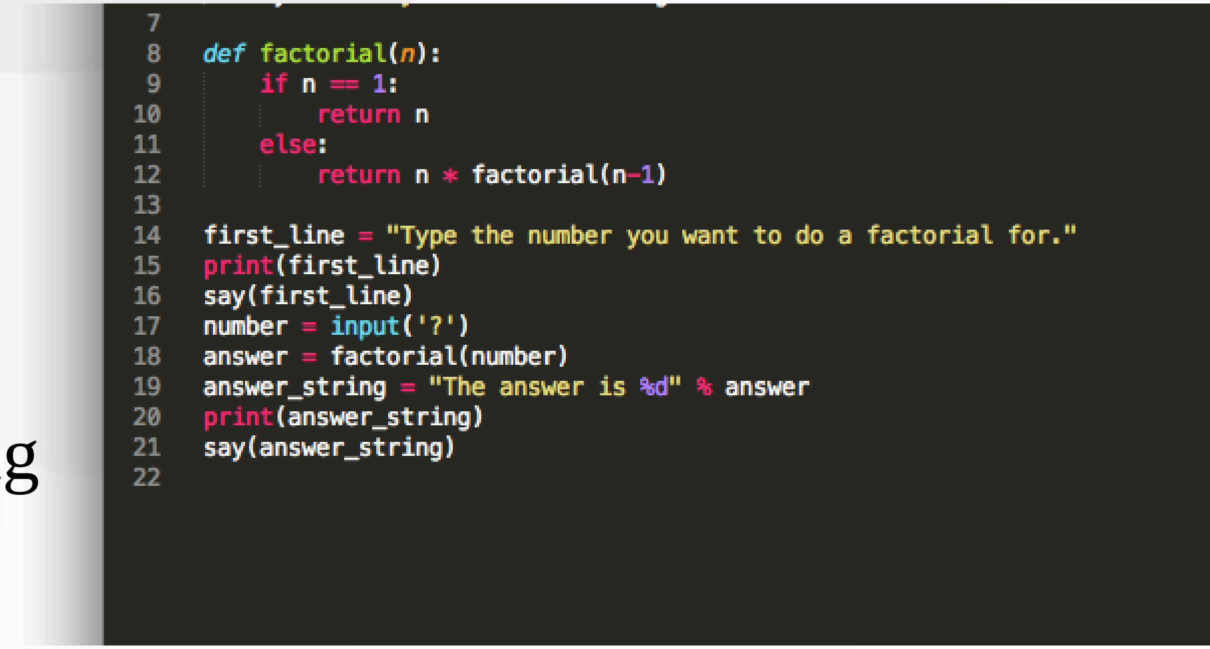
Control Flow

- How to design a program?

Computer Program

A computer program is **a collection of instructions** that can be executed by a computer to perform a specific task.

- usually written by a computer programmer in a high-level programming language (Python， Java， C++).
- is human-readable form of source code
- a compiler or assembler is required to generate machine code—a form consisting of instructions that the computer can directly execute.



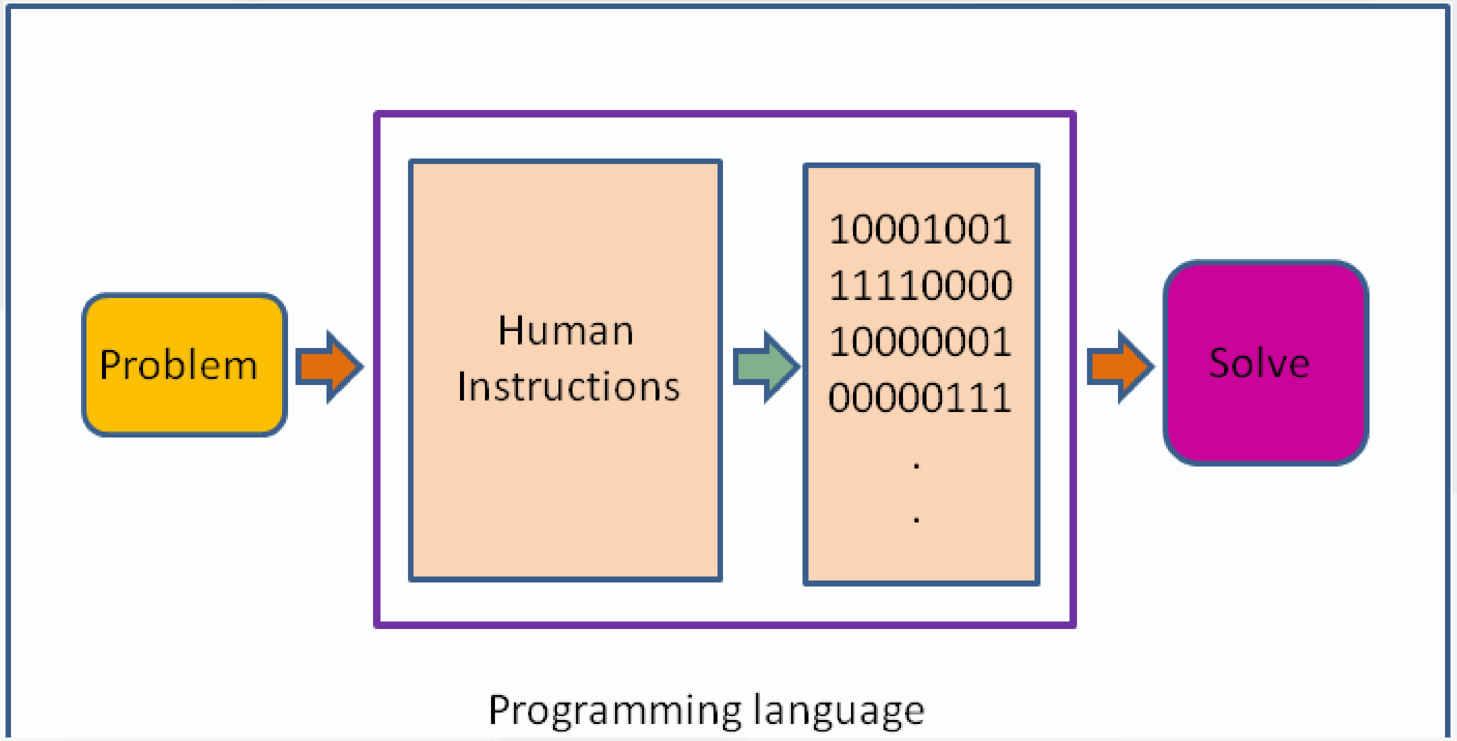
```

def factorial(n):
    if n == 1:
        return n
    else:
        return n * factorial(n-1)

first_line = "Type the number you want to do a factorial for."
print(first_line)
say(first_line)
number = input('?')
answer = factorial(number)
answer_string = "The answer is %d" % answer
print(answer_string)
say(answer_string)

```

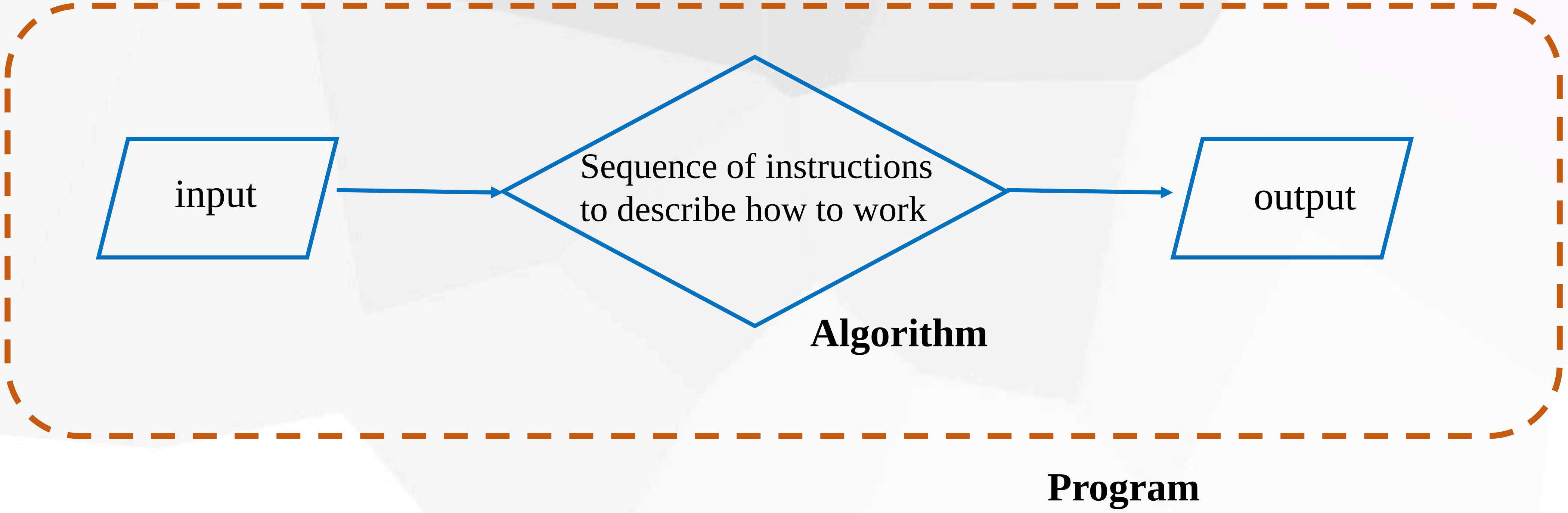
Computer Program



A computer program is a **executable software that runs on a computer**, helping to solve problems, which designed by human instruction, and executed by machine code

Algorithm in Program

Algorithm as a recipe that describes the exact steps needed for the computer to solve a problem



Algorithm in Program

Algorithms, in general can be designed as

- Flow charts-visually present the design
- Pseudocode-describe the steps with human language
- Program code-translate into program instructions

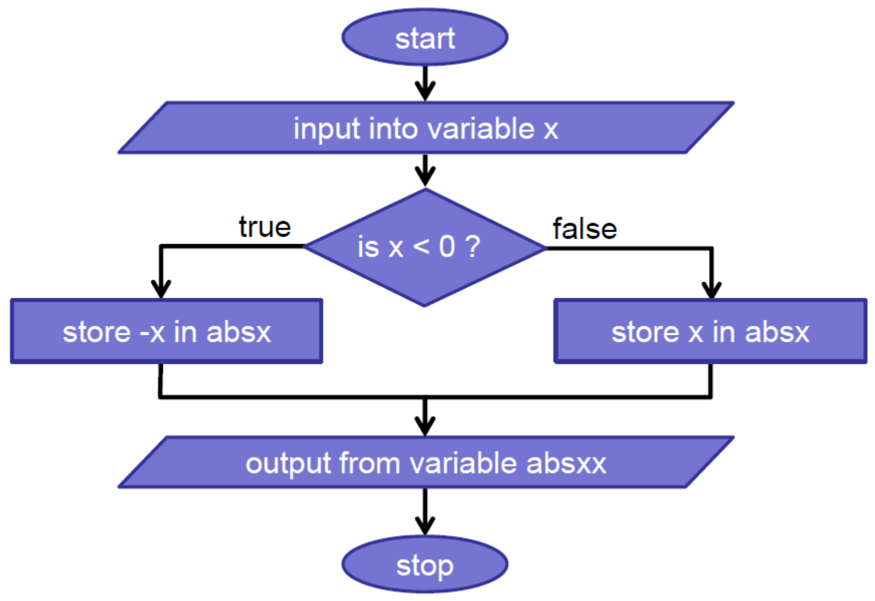
Algorithm in Program

Flowcharts

- Allow organizing control flow more visually.
- *Check the path of the control based on input.*
- Change the path based on input.

Example

- Read a number.
- If the number is positive, then store the number as is.
- If the number is negative, store the negative of the number

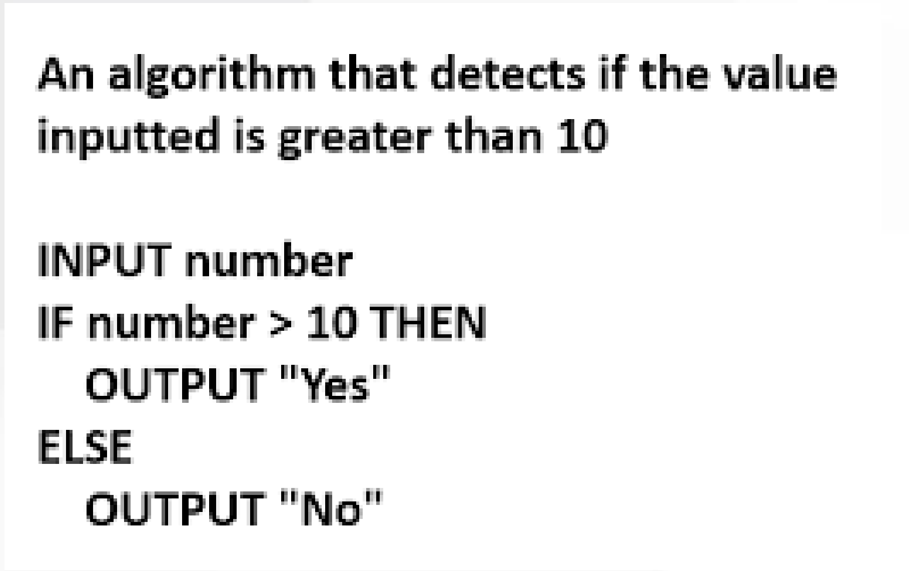


- Could you draw a flowchart for registering a new semester?

Algorithm in Program

Pseudo code

- Pseudocode is an informal way of programming description, understood by the programmers of all types
- does not require any strict programming language syntax
- used for creating an outline or a rough draft of a program.
- enables the programmer to concentrate only on the algorithm part of the code development.

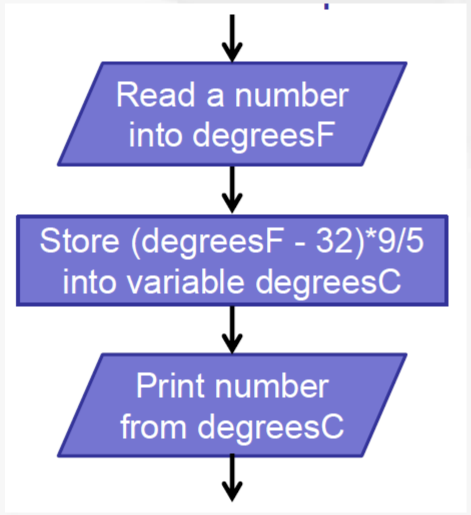


An algorithm that detects if the value inputted is greater than 10

```
INPUT number
IF number > 10 THEN
   OUTPUT "Yes"
ELSE
   OUTPUT "No"
```

Algorithm in Program

With flow chart and Pseudo code, the algorithms can be achieved by any program language.


Read a number into degreesF
Store (degreesF - 32)*9/5 into variable degreesC
Print number from degreesC

```
READ degF
COMPUTE degC AS (degF – 32)*5/9
DISPLAY degC
```

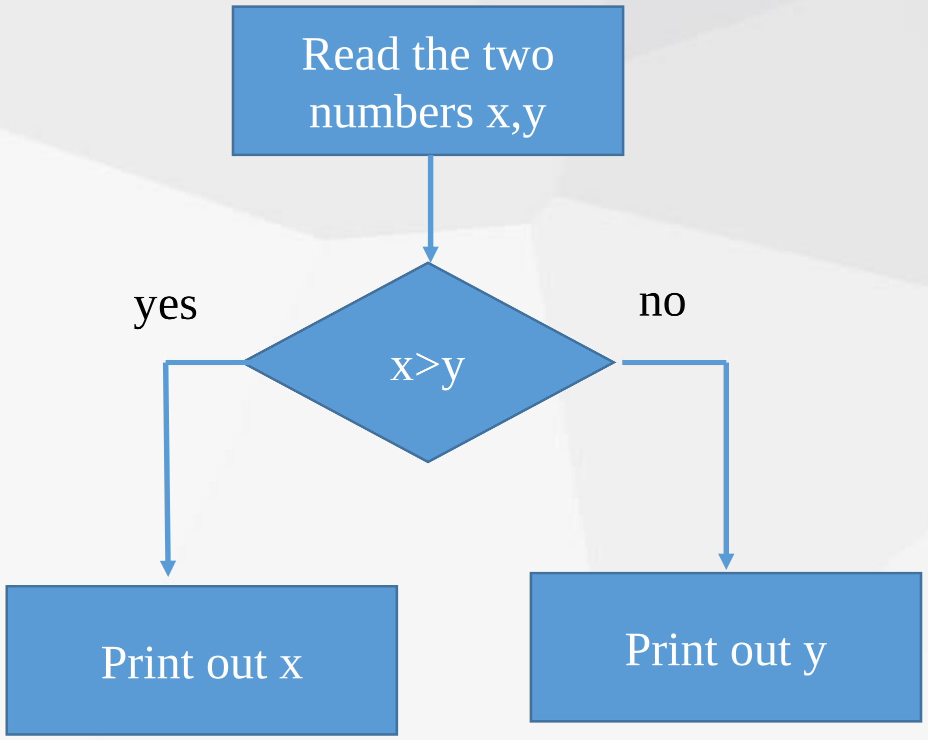
```
degF  = input(“F-temperature? ”)
degC = (degF – 32)*5/9
print(degC)
```

Flow chart $\longrightarrow$ Pseudo code $\longrightarrow$ Python code

Example Program 1

Problem. Write a program to find the larger of two numbers.

draw a flowchart



Algorithm.

```
READ firstNumber x,
READ secondNumber y,
IF x > y THEN DISPLAY firstNumber
ELSE
    DISPLAY secondNumber
ENDIF
```

Example Program 1

Problem. Write a program to find the larger of two numbers.

Python code:

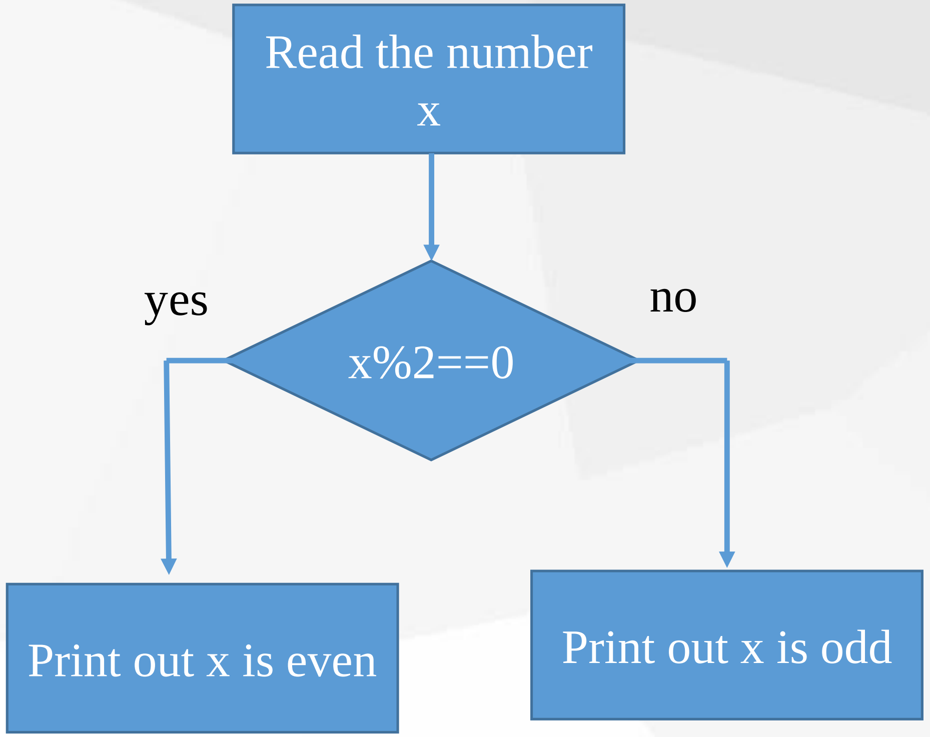
```
x=eval(input("input the first number"))
y=eval(input("input the second number"))
if x>y:
    print("the larger one is ", x)
else:
    print("the larger one is ", y)
```

```
input the first number 3
input the second number 5
the larger one is  5
```

Example Program 2

Problem. Determine if an integer is even or odd.

draw a flowchart



ALGORITHM:

```
READ number
IF number % 2 IS 0   THEN
   DISPLAY even
ELSE
  DISPLAY odd
ENDIF
```

主观题 10分

设置

Problem. Determine if an integer is even or odd.

Python code:

```
x=eval(input("input the number"))
if (x%2==0):
    print("the number is even")
else:
    print("the number is odd ")
```

正常使用主观题需2.0以上版本雨课堂

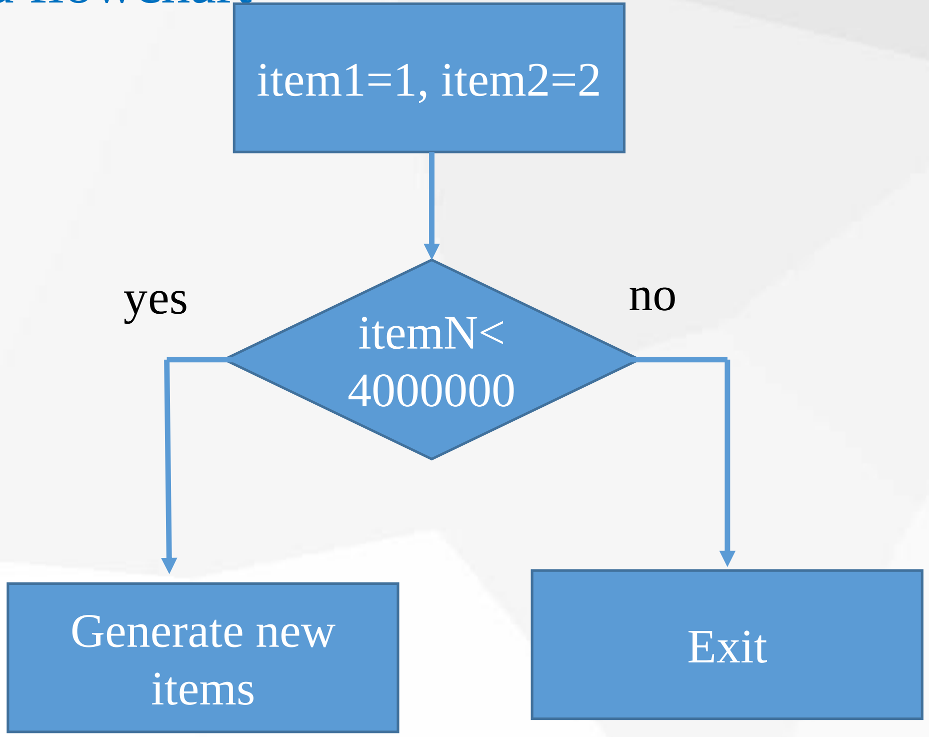
作答

Example Program 3

Problem. Print out the Fibonacci sequence with four million (4000000), by starting with 1 and 2.

Fibonacci sequence is generated by adding the previous two terms, like the first 10 terms will be: 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...

draw a flowchart



ALGORITHM:

```
Initialize total=0,
Initialize item 1=1 and item 2=2

While the new item < 4000000

    renew the item 1 and item 2
     item 1=item2
     item2=item1+item2
```

主观题 10分

设置

Problem. By considering the terms in the Fibonacci sequence whose values do not exceed four million (4000000), print the Fibonacci sequence .

.

```
total = 0
f1, f2 = 1, 2
print(f1,f2,sep='\n')
while f1 < 4000000:
    f1, f2 = f2, f1 + f2
    print(f2)
```

```
1
2
3
5
8
13
21
34
55
89
144
233
377
610
987
1597
2584
4181
6765
10946
17711
28657
46368
75025
121393
196418
317811
514229
832040
1346269
2178309
3524578
```

作答

THANKS FOR YOUR ATTENTION!