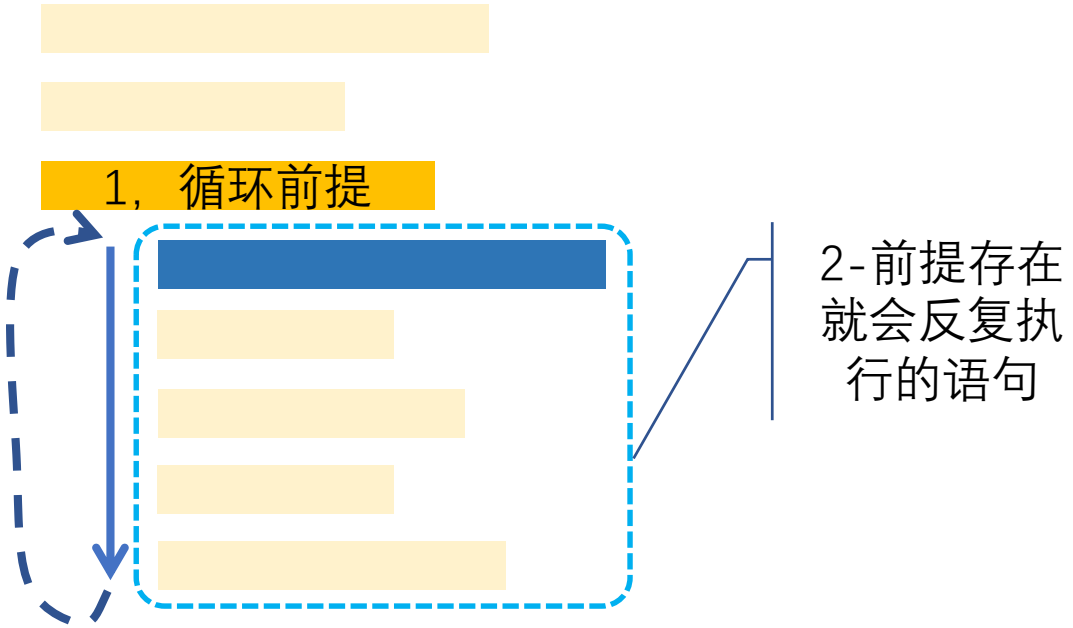
# Python程序设计04-基本控制流程

北京大学 陈斌

2019.03.18

# 目录

- 重复：循环结构
- 迭代循环：for语句
- 常用的迭代数据集
- 绘制数学曲线




循环结构的两个基本要素

# Python语言的几个要件

## 数据对象和组织

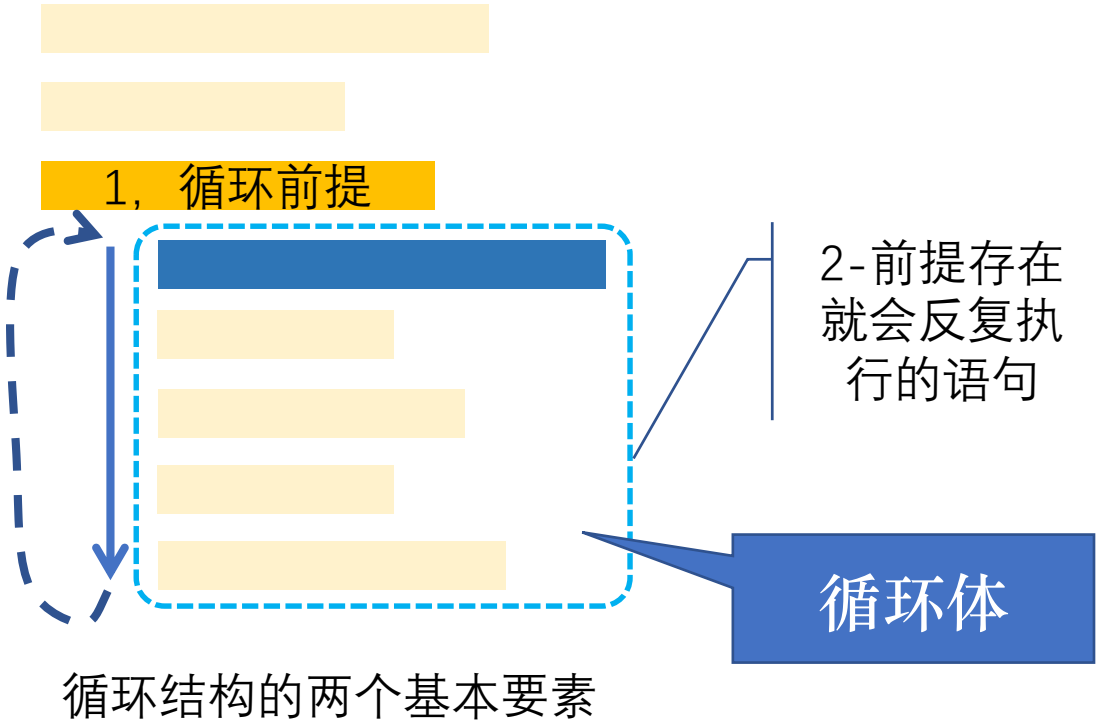
- 对现实世界实体和概念的抽象
- 分为简单类型和容器类型
- 简单类型用来表示值
  - 整数int、浮点数float、复数complex、逻辑值bool、字符串str
- 容器类型用来组织这些值
  - 列表list、元组tuple、集合set、字典dict
- 数据类型之间几乎都可以转换

## 赋值和控制流

- 对现实世界处理和过程的抽象
- 分为运算语句和控制流语句
- 运算语句用来实现处理与暂存
  - 表达式计算、函数调用、赋值
- 控制流语句用来组织语句描述过程
  - 顺序、条件分支、循环
- 定义语句也用来组织语句，描述一个包含一系列处理过程的计算单元
  - 函数定义、类定义

# 重复：循环结构（loop）

- 我们需要让计算机反复做设定的任务
- 又能在该停止的时候自动停止重复
- 循环结构具有两个要素
  - 一个循环前提
  - 一组重复执行的语句（循环体）
- 只要循环前提成立，循环体就会被反复执行



循环结构的两个基本要素

# 迭代循环：for语句

- 迭代循环语句：for语句
- 循环前提：
  - 一个（或一组）循环变量
  - 一个数据对象集
- for语句每次从对象集中取出一个数据对象，赋值给循环变量
  - 如果能取到，就执行一次循环体
    - 循环体中可以使用循环变量
  - 如果取完了，就退出循环

forst.py - /Users/chenbin/Documents/教学项目/北大附

```
s = 0
for i in range(1, 101):
    s = s + i
print("sum:1..100:", s)

a, b, c = 10, -5, 0.5
for x in [-25, 0, 10, 35, 100]:
    y = a * x * x + b * x + c
    print (f"f({x})={y}")

for name in ["Tom", "Jerry", "张三"]:
    print("Hello!", name)
```

# 常用的数据集：range函数

- range函数可以产生连续整数构成的数据集
- range(end)
  - [0, end)
- range(start, end)
  - [start, end)
- range(start, end, step)
  - [start, end) 步长step
  - 如果step小于0则反向取

range()函数产生一个连续整数的数据集

range(end)

range(start, end)

range(start, end, step)

```
1   print("range:5")
2   for i in range(5):
3       print(i)
4
5   print("range:1,5,2")
6   for i in range(1, 5, 2):
7       print(i)
8
9   print("range:4,1,-1")
10  for n in range(4, 1, -1):
11      print(n)
```

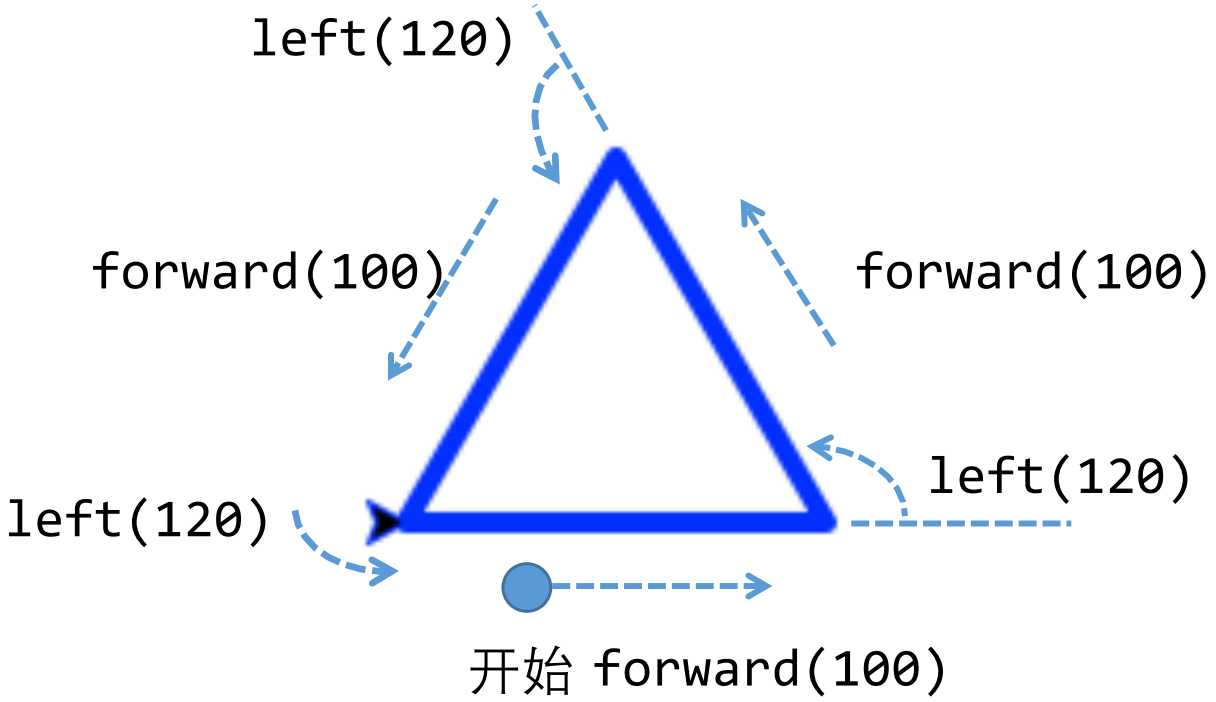
# 常用的数据集：列表list

- 列表是一种容器数据类型，可以包容多个数据对象
- 整数/浮点数列表
  - [1, 3, 5, 35, -10]
  - [1.23, 34.5, 10.0, 245.7]
- 字符串列表
  - ["Tim", "Jay", "Mary"]
- 混合列表
  - ["Hello", True, 12, 34.56]

```
for name in ["Tom", "Jerry", "张三"]:
    print("Hello!", name)

for n in [1, 3, 5, 35, -10]:
    print(f"{n}^2=", n * n)

m = 1
for f in [1.23, 34.5, 10.0, 245.7]:
    m *= f
print("product:", m)

for k in [12, 30, 8, 10, 9]:
    print(f"{k:02d}>", "#"*k)
```

# 练习：循环语句的应用

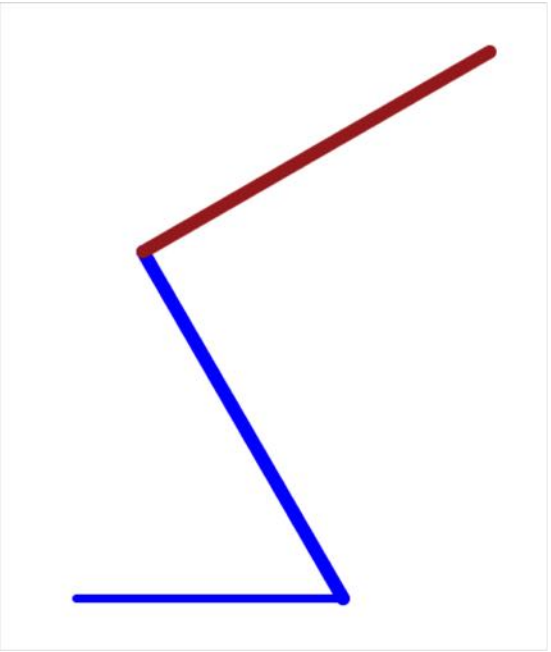
- 如何用循环语句画出等边三角形?
- 如何用循环语句画出一个4种颜色边的正方形?



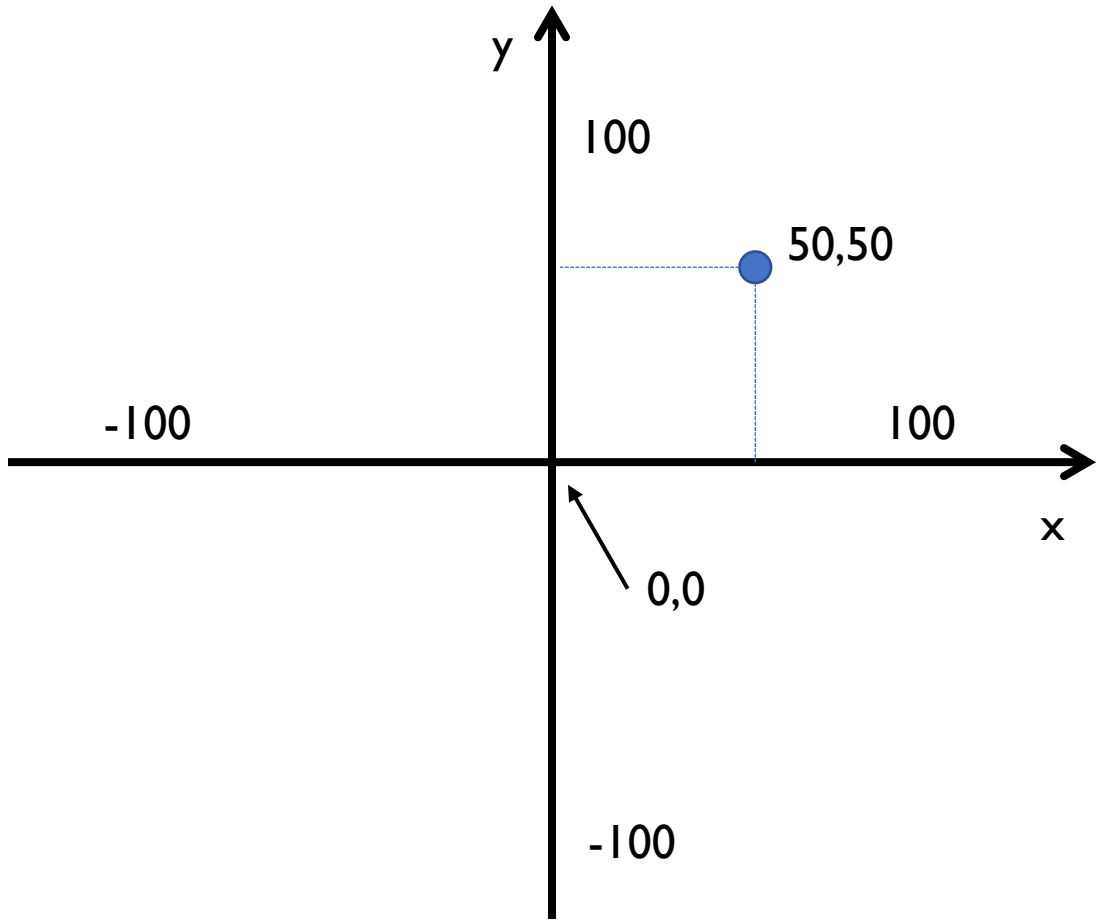
# 作图程序模版

- 首先，导入turtle模块
- 然后，生成一只海龟
  - 可以做一些初始化设定
- 程序主体：用作图语句绘图
- 最后结束作图
  - 可选隐藏海龟：t.hideturtle()

```
1   # 1. 导入海龟模块
2   import turtle
3
4   # 2. 生成一只海龟，做一些设定
5   t = turtle.Turtle()
6   t.color("blue")
7   t.pensize(3)
8
9   # 3. 用海龟作图
10  t.forward(100)
11  t.right(60)
12  t.pensize(5)
13  t.backward(150)
14  t.left(90)
15  t.color("brown")
16  t.forward(150)
17
18  # 4. 结束作图
19  t.hideturtle()
20  turtle.done()
```



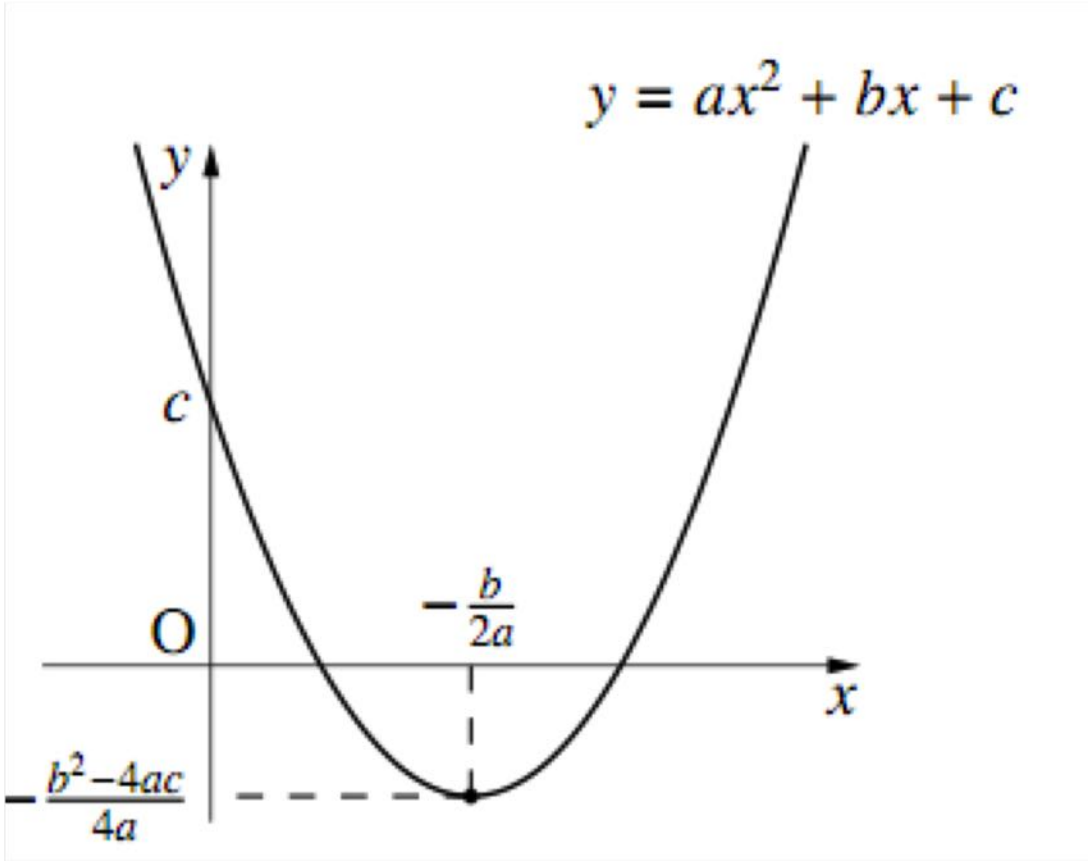
# 让海龟直接去到某个位置goto(x,y)




- 最开始海龟在 (0,0)
- 可以用goto让海龟去到任何地方
- 用position()获取海龟当前位置
- 如果不希望留下轨迹，则需要在goto之前调用penup()

# 平面直角坐标系曲线绘制

- 如何用循环语句绘制数学曲线？
  - y=ax+b、$y=ax^2+bx+c$、y=sin(x)
- 一般步骤
  - 估计x,y的范围
  - 设定坐标系：左下角/右上角坐标
  - 画出坐标轴（可选：标注公式）
  - 迭代循环x，计算y
  - goto(x,y)将点连接起来
  - 可以叠加多条曲线

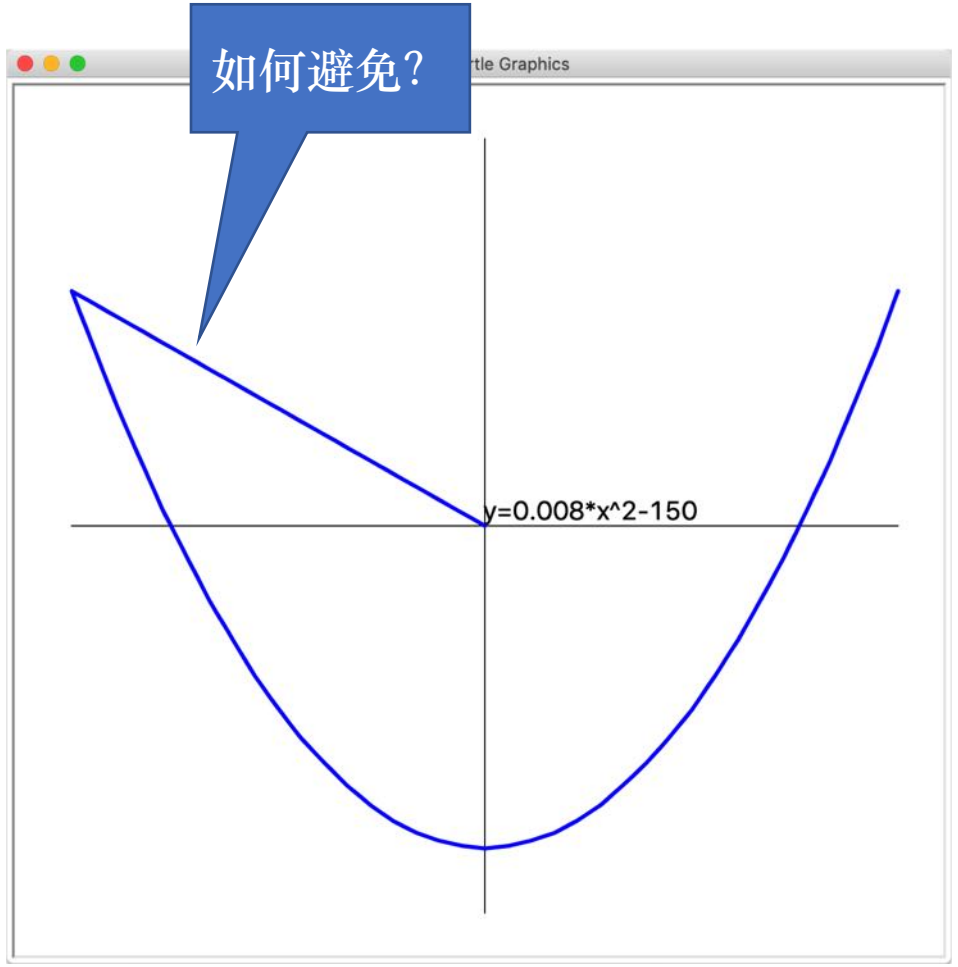



北京大学 陈斌 gischen@pku.edu.cn

# 示例：绘制数学曲线

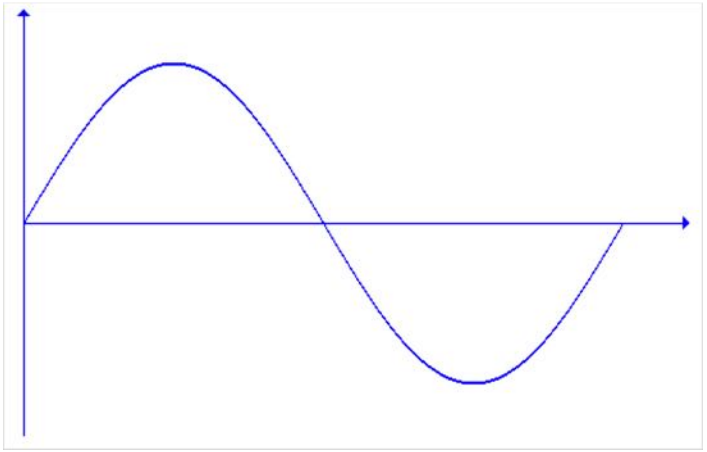
```
1   # 1. 导入海龟模块
2   import turtle
3
4   # 2. 生成一只海龟，做一些设定
5   t = turtle.Turtle()
6   # 设定坐标系：左下角x,y；右上角x,y
7   turtle.setworldcoordinates(-200, -200, 200, 200)
8
9   # 3. 用海龟作图
10  t.penup()
11  t.goto(-180, 0)
12  t.pendown()
13  t.goto(180, 0)
14  t.penup()
15  t.goto(0, -180)
16  t.pendown()
17  t.goto(0, 180)
18  t.goto(0, 0)
19  t.write("y=0.008*x^2-150", font=("consolas", 20, "normal"))
20  t.pencolor("blue")
21  t.pensize(3)
22  for x in range(-180, 181, 10):
23      y = 0.008 * x * x - 150
24      t.goto(x, y)
25
26  # 4. 结束作图
27  t.hideturtle()
28  turtle.done()
```

如果x是小数怎么办?



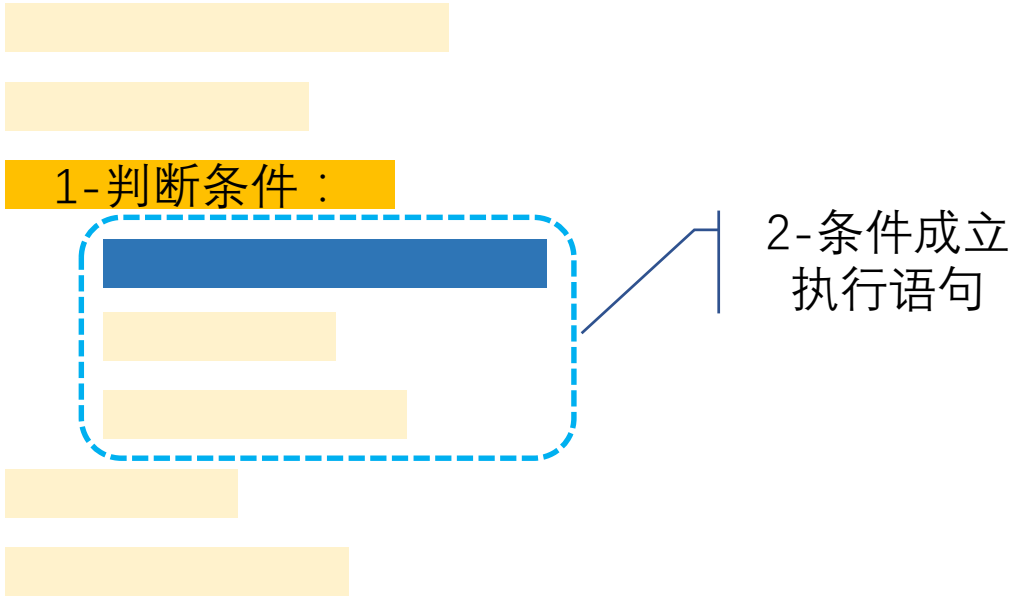
# 随堂作业：绘制三角函数曲线

- 写一个程序tri.py
- 提交希悦
- 叠加绘制下面3个函数
  - 绿色：y=sin(x)
  - 红色：y=cos(x)
  - 蓝色：y=2cos(2x)
  - x的范围是-2pi～2pi



# 条件分支结构：if语句

- 让计算机能够自动根据当前的状况来决定执行哪些语句
- 条件分支结构的2个要素
  - 判断条件
  - 一组语句
- if语句首先计算判断条件
  - 如果得到True，就执行这组语句
  - 否则，不执行




条件分支结构两个基本要素

```
# 判断偶数
n = int(input("n="))
print("Your number is", n)
if n % 2 == 0:
    print("It's a even number!")
```

# if语句的附加要素：elif和else

- if语句可以附加两个子句
- else子句可以指定在判断条件不成立的时候，要执行的一组语句
- elif子句可以在判断条件不成立的时候，再继续判断另一个条件，相当于else：if

```
# 计算x1和x2之间的距离
x1 = int(input("x1="))
x2 = int(input("x2="))
if x1 > x2:
    d = x1 - x2
else:
    d = x2 - x1
print("distance=", d)
```

```
# 判断年龄
age = int(input("age="))
print("年龄: ", age)
if 0 <= age <= 6:
    print("童年")
elif 7 <= age <= 17:
    print("少年")
elif 18 <= age <= 40:
    print("青年")
elif 41 <= age <= 65:
    print("中年")
else:
    print("老年")
```

# 随堂作业：判断三角形/找最大数

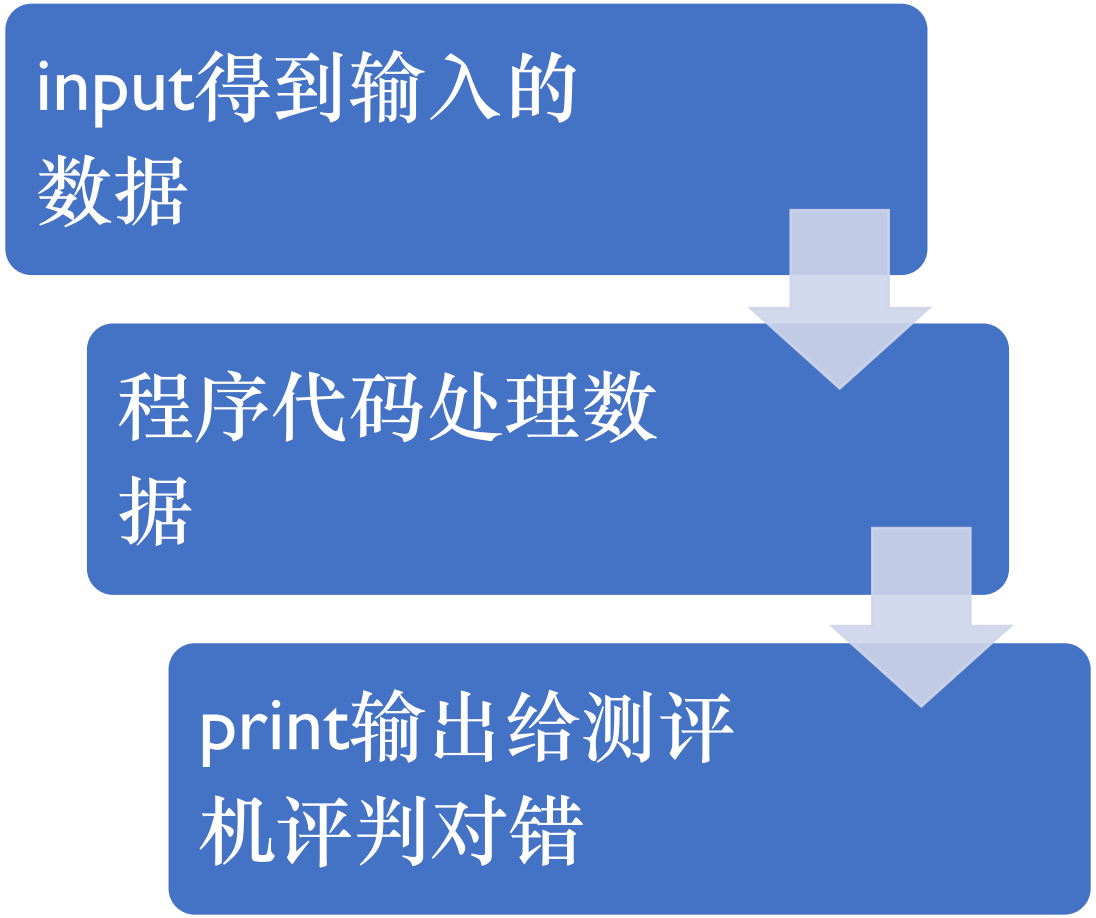
- 程序1（judge3.py)
  - 输入a,b,c三个整数
  - 输出这三个长度的边是否可以构成三角形
- 程序2（fmax.py)
  - 输入若干个数，用空格隔开
  - 输出这些数中的最大数

```
# 获取输入一行用空格隔开的整数
numbers = input("some numbers:").split()
numbers = list(map(int, numbers))
print(numbers)
```

```
# 简写
numbers = list(map(int, input().split()))
```

# 什么是OJ?

- Online Judge，在线测评
- 会有若干组输入输出来判断程序代码是否正确
- 每组输入输出
  - 输入：由input读入
  - 输出：print输出
- 测评机只判断输出的字符串跟标准答案是否相等
  - 输出分行、分隔符号、小数点位数、大小写等等都要完全一致



# 关于Online Judge中Python代码的技巧

- 提示：不要在input里加任何提示符的参数！
- 读入数据：一行就一个值
  - `astr = input()`
  - `n = int(input())`
  - `f = float(input())`
- 读入数据：一行多个整数值
  - `alist = list(map(int, input().split()))`

北京大学 陈斌 gischen@pku.edu.cn

# 关于Online Judge中Python代码的技巧

- 输出数据：一行就一个值
  - `print(n)`
  - `print("%.2f", f)  # 小数点后两位`
- 输出数据：一行多个值
  - `print(m, n, i)  # 三个整数`
  - `print(" ".join(map(str, alist)))`

北京大学 陈斌 gischen@pku.edu.cn

# 【H4】完成OJ题：基本表达式

- 请到如下网址完成练习题：
  - https://vijos.org/d/pkuchenbin/training/5c8e53d5f413620934d099a8