一、变量

顾名思义，可以改变的量。相当于挖了个坑，可以往里面赛各种东西。

```
# 将 100 塞给 a
a = 100
print(a)  # 输出: 100

# 将 'abc' 塞给 a (a原来的值被替代掉)
a = 'abc'
print(a)  # 输出: abc
```

二、分支

```
a = int(input())
b = int(input())
if a > b:
    print(1)  # a > b 才会执行
else:
    print(2)  # 不满足 a > b 才会执行
```

```
a = int(input())
b = int(input())
if a % 2 == 0 and b % 2 == 0:
    print('都是偶数')
# 不满足 a % 2 == 0 and b % 2 == 0 即 不全为偶数，也就是只有一个或者没有
elif a % 2 == 0:  # 因为只可能有一个或者没有，如果a是偶数那么就只有它是偶数
    print('只有 a 是偶数')
elif b % 2 == 0:
    print('只有 a 是偶数')
else:  # 上面都不满足，即a和b都不是偶数
    print('都不是偶数')
```

三、循环

```
n = 7
while n > 0:
    print(n)
    n = n - 1
```

while 循环：满足条件则执行一次内部语句，执行完毕再返回检查，还满足就再执行，以此反复循环。

```python
for i in [0, 2, 5, 6, 9]:
    print(i, end=' ')  # end=' ' 的意思是每次打印以 ' ' 结尾（默认是换行）
# 输出：0 2 5 6 9

# range 为左闭右开
for i in range(0, 9):
    print(i, end=' ')  # end=' ' 的意思是每次打印以 ' ' 结尾（默认是换行）
# 输出：0 1 2 3 4 5 6 7 8
```

四、列表

用 `[]` 括起来，由多个元素组成，每个元素之间以逗号隔开。支持添加元素、删除元素、修改元素、获取元素。

（暂时先知道 `a.append(x)` 向尾部追加 `x`， `a[i]` 获取下标为 `i` 的元素，`len(a)` 获取 `a` 的长度就好）

```python
a = []  # 空列表
a = [2, 4]  # 列表 [2, 4] 赋给 a

# 在a中的列表尾部添加6
a.append(6)  # a变为：[2, 4, 6]

# 打印a中列表的长度
print(len(a))  # 输出：3

# 打印下标为1的元素
print(a[1])  # 输出：4
```

五、例

例0：交换两个数 再交换两个数（不同方法）

```python
a, b = 1, 2

print(a, b)
t = a
a = b
b = t
print(a, b)
a, b = b, a
print(a, b)
```

例1：遍历列表

遍历，遍 - 统统地，历 - 看一遍。
我在这写两种方法：

```
# 令 i 分别为 a 中所有元素，循环
for i in a:
    print(i, end=' ')
# 输出：2 4 6

# 令 i 为 0到a的大小，循环
for i in range(0, len(a)):
    print(a[i], end=' ')
# 输出：2 4 6
```

例2：输入100个数存在列表中并打印出来

```
a = []
for i in range(100):  # range中只写一个数，那就是相当于写range(0, 这个数)
    a.append(input())
print(a)
```

拓展：生成100个随机数存在列表中并打印出来
使用 `random` 包中的 `randint()` 函数来生成随机整数。

```
import random
a = []
for i in range(100):
 a.append(random.randint())
print(a)
```

更进一步生成n个数：

```
import random
n = eval(input())
a = []

for i in range(n):
 a.append(random.randint())
print(a)
```

例3：找到n个数中的最大值并输出

```
import random
n = eval(input())
a = []

for i in range(n):
```

```
    a.append(random.randint())

ans = a[0]
for i in a:
    if ans < i:
        ans = i
print(ans)

# 或者这么写:
ans = a[0]
for i in range(len(a)):
    if ans < a[i]:
        ans = a[i]
print(ans)
```

例4：将n个数中的最大数挪到最右侧

```
import random
n = eval(input())
a = []

for i in range(n):
    a.append(random.randint())

for i in range(len(a)-1):
    if a[i] > a[i+1]:
        a[i], a[i+1] = a[i+1], a[i]
print(ans)
```

例5：排序n个数

```
import random
n = eval(input())
a = []

for i in range(n):
    a.append(random.randint())

for i in range(len(a)):
    for j in range(len(a)-1):
        if a[j] > a[j+1]:
            a[j], a[j+1] = a[j+1], a[j]
print(ans)
```

拓展：计时

`time` 包中的 `time()` 函数可以获取当前的 **时间戳**（从1970年1月1日0点0分到现在的秒数）

```python
import time
print(time.time())
```

那么就可以这样来计算程序运行经过的时间：

```python
import random
import time

random.seed(7777)  # 设置随机数种子确保每次运行生成的列表相同（如不设定，种子值取决于当前时间，所以每一次都不一样）（见下方）

a = []
for i in range(10000):
    a.append(random.randint(0, 1000))

t1 = time.time()
for i in range(len(a)):
    for j in range(len(a)-1):
        if a[j] > a[j+1]:
            a[j], a[j+1] = a[j+1], a[j]
t2 = time.time()
print(f"耗时：{t2-t1}秒")
```

> 耗时：13.917094230651855秒

这里为了实验的严谨性，通过 `random.seed(7777)` 设定随机数种子为 `7777` 来保证每次运行程序生成的随机数列表相同。

> 计算机中的随机数为伪随机数，是使用一种特殊的算法，由一个起始值（被称作种子seed）进行迭代计算算出的值。
> 故如果seed相同，每次运行生成的第i个随机数都是相同的（第i次调用 randint() 产生的第i个随机数由第i次迭代产生，初始值相同那么显然迭代过程也相同）

其实我们发现，每排序完一轮，就可以少比较一次，因为上一次排序已经确定了一个数的位置，就不用管它了

```python
import random
import time

random.seed(7777)

a = []
for i in range(10000):
    a.append(random.randint(0, 1000))

t1 = time.time()
for i in range(len(a)):
    for j in range(len(a)-i-1):  # 在这里进行优化，可以少做一些没有意义的比较
        if a[j] > a[j+1]:
```

```
            a[j], a[j+1] = a[j+1], a[j]
t2 = time.time()
print(f"耗时：{t2-t1}秒")
```

> 耗时：9.86142349243164秒

可以发现快了很多

其实可以用 `a.sort()` 来直接排序 a 这个列表，但是为了巩固循环等知识，在开始阶段应该亲自尝试自己实现，虽然上述 **冒泡排序** 算法效率不高，但是对锻炼自己对基础语句的理解与把握很有帮助。
你猜，`a.sort()` 要排多久？

```
import random
import time

random.seed(7777)

a = []
for i in range(10000):
    a.append(random.randint(0, 1000))

t1 = time.time()
a.sort()
t2 = time.time()
print(f"耗时：{t2-t1}秒")
```

> 耗时：0.0009996891021728516秒