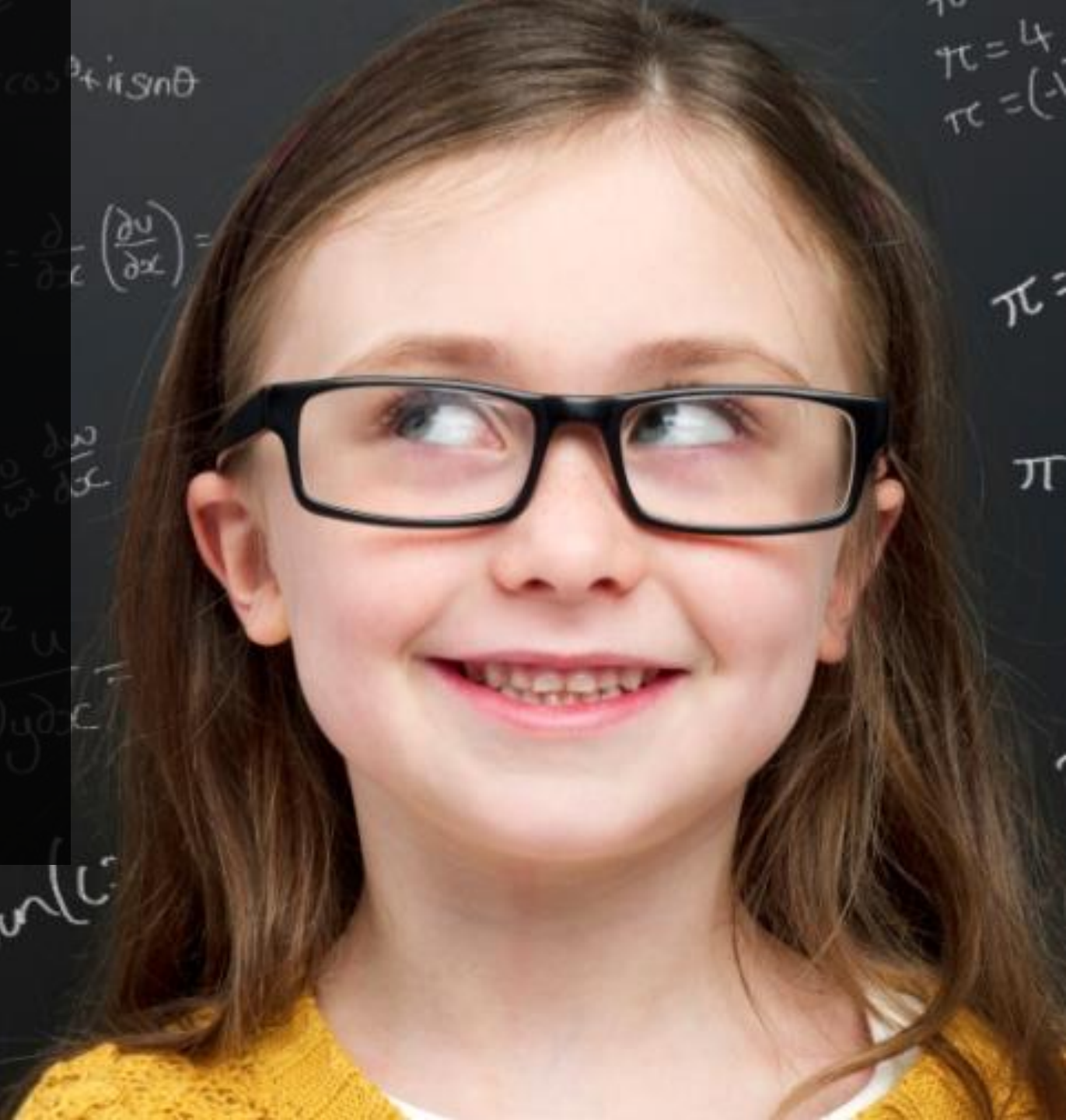


# 亲子算法课 (3) ——开始写第 一个程序

作者：叶蒙蒙





# 编程和编程语言



# 什么是编程

用计算机能看懂的  
形式描述算法

——把算法写给计算机看





Assembly Language

Machine Language

```
mov ecx, ebx  
mov esp, edx  
mov edx, r9d  
mov rax, rdx
```

Assembler + Linker

```
100101011001  
010011111011  
111010101101  
01010101010
```

Programmer

Processor



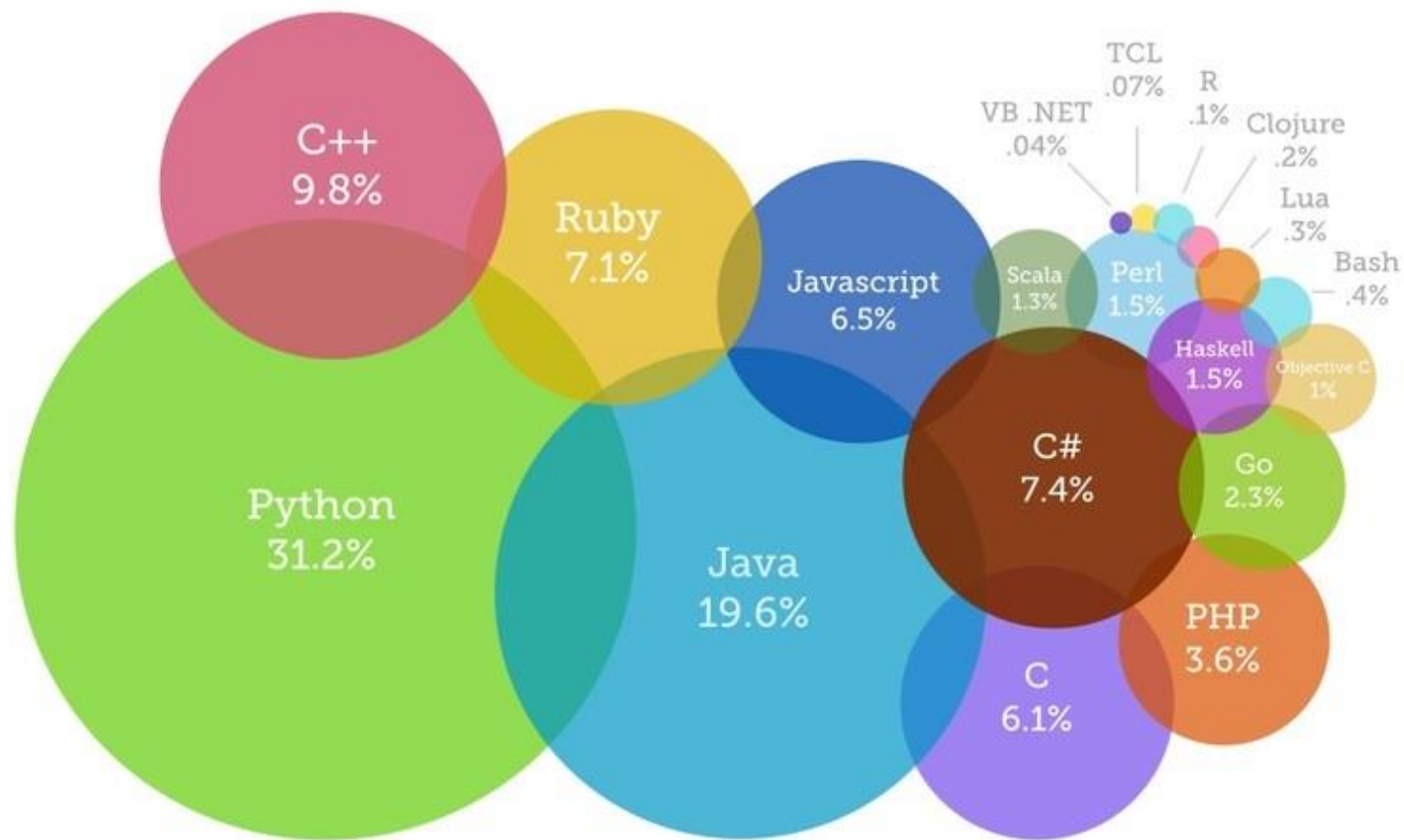
# 什么是编程语言

- 用来编写计算机程序的语言
- 编程语言的发展
  - 0-1代码
  - 汇编语言
  - 高级语言



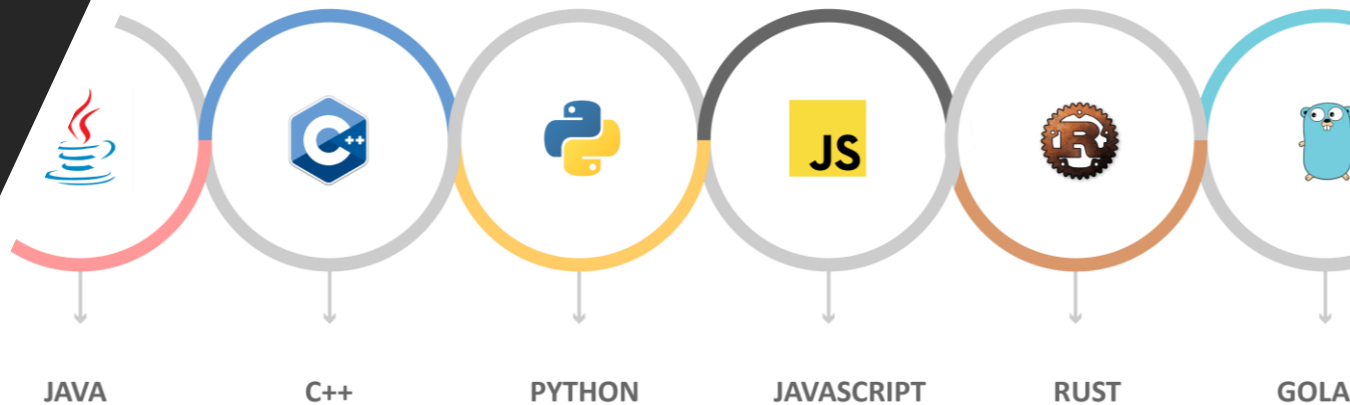
## 常见的编程语言

Most Popular Coding Languages of 2018



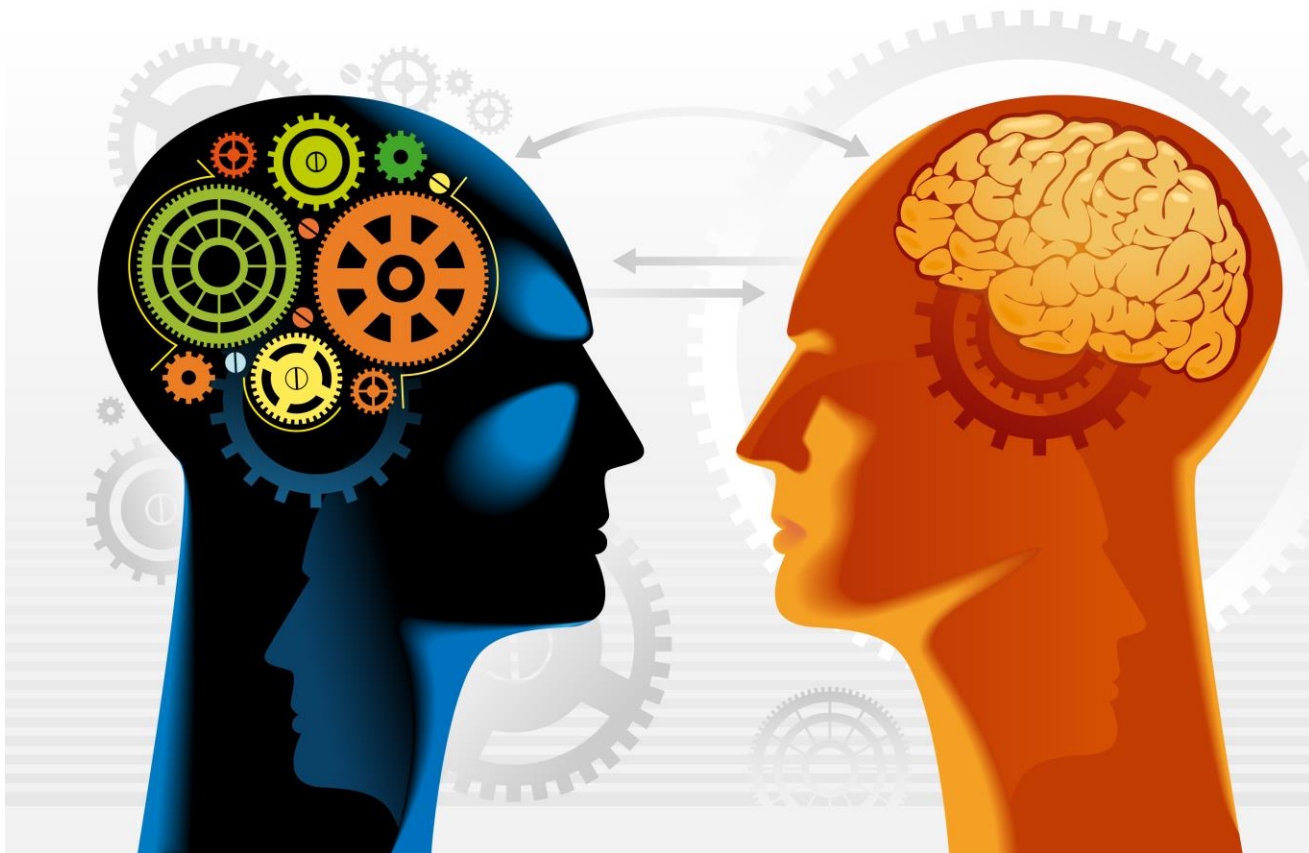
# 自然语言 vs 编程语言

- 词汇 – 关键字、操作符、表达
- 语法 – 句法 (Syntax)
- 习惯用法 (俗语、成语等) – 库函数
- 段落 – 代码块 (block)
- 文章 – 程序



# 自然语言和编程语言哪个更难学？

- 当然是自然语言
- 学习时间的区别
  - 自然语言->几年
  - 编程语言->几个月
- 编程语言的好学之处
  - 极小的“词汇量”
  - 超级严格的“语法”
- 编程语言的难学之处
  - 对逻辑的表达——算法！  
算法！ 算法！





```
# End onClick function  
# End ZoomToSelectedFeatures class  
class ZoomToSelectedFeatures(object):  
    '''Implementation of the zoom-to button.'''  
    # Implementation of onClick method of Button's class  
    def onClick(self):  
        # Get the current map document and data frame  
        mxd = arcpy.mapping.MapDocument('current')  
        df = arcpy.mapping.ListDataFrames(mxd)[0]  
        df.zoomToSelectedFeatures()  
    # End onClick function  
# End ZoomToSelectedFeatures class
```

# Python语言

- <https://www.python.org/>
- 解释型语言
  - 无须编译
  - 各种平台通用
  - 速度慢
- 简洁、可读性强
- 支持多种编程范式
  - 面向对象
  - 命令式
  - 函数式
  - 过程式
- 动态类型语言 (\*)
- 强类型语言 (\*)



# Python语言是缩进 (Indentation) 敏感的

- 每一行的缩进（最开始空几个格）非常重要！！！！
- Python的块（block）通过缩进来区分
- 每4个空格的缩进表示了一个层次

Block 1

Block 2

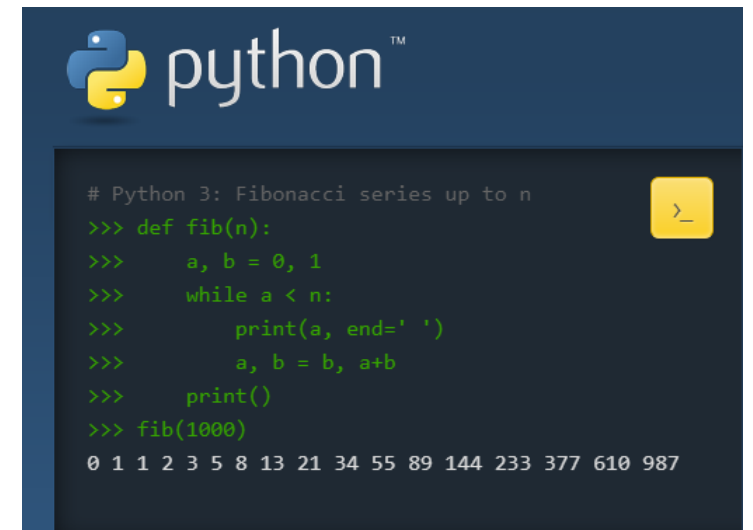
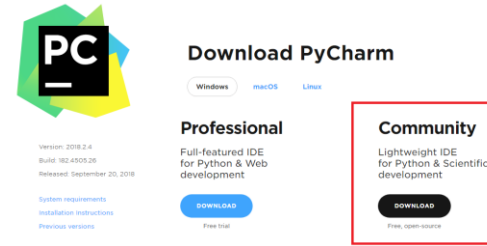
Block 3

Block 2, continuation

Block 1, continuation

# 在编写Python程序之前需要安装的软件

- Python运行环境
  - 我们的选择：Python3.6
  - <https://www.python.org/downloads/release/python-366/>
- (\*) IDE
  - 我们的选择：PyCharm Community
  - <https://www.jetbrains.com/pycharm/download/download-thanks.html?platform=windows&code=GCC>



# 编写第一个程序





元素 (Element)	语句 (Statement)	块 (Block)	函数 (Function) /过程 (Procedure)	模块 (Module)	程序
关键字 (Keyword) 表达 (Expression) <ul style="list-style-type: none"><li>•标识符 ( identifier)</li><li>•字面量 (literal)</li><li>•操作符 (operator)</li></ul>	一个语句一般包含一个或多个 i) 表达, 或者ii) 关键字和表达 一般一行就是一个语句。 *语句太长也可以换行, 但需要加换行符	一个块包含一个或多个语句	包含一到多个块	一到多个函数/过程	一到多个模块

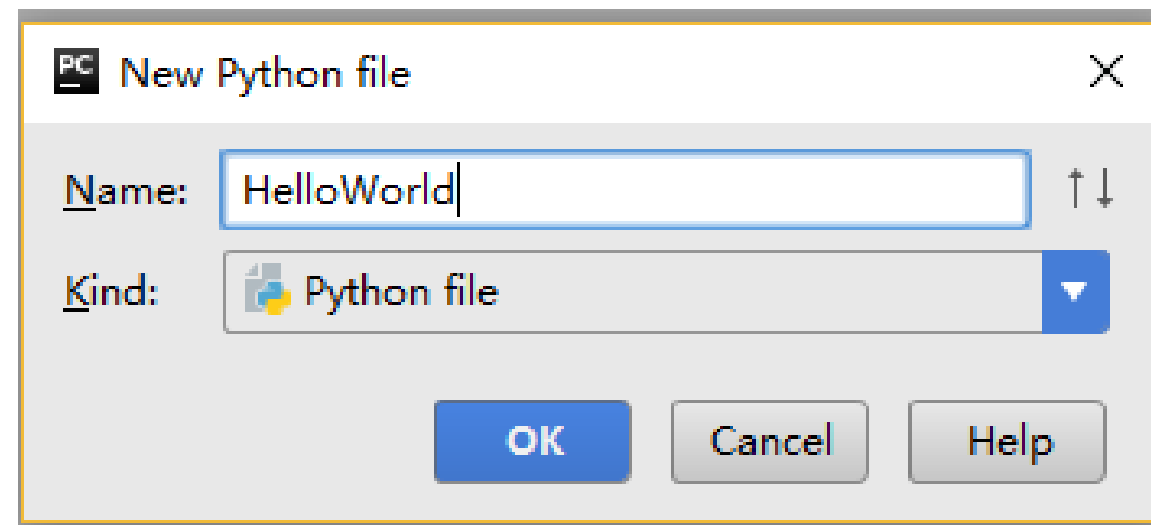
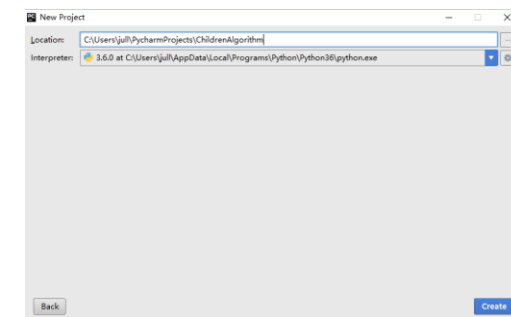
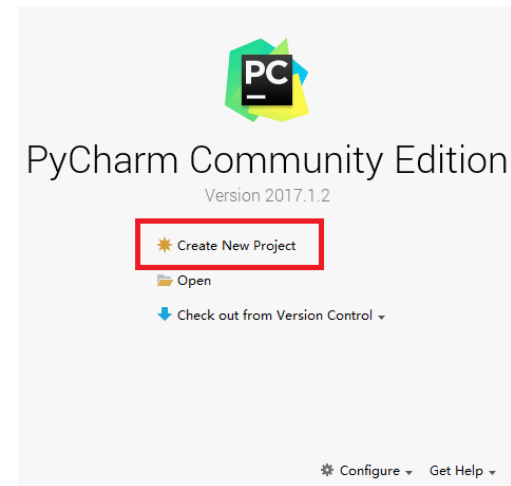
# 计算机程序的构成

# 第一个Python程序 ——Hello World (1)

- 打开PyCharm，创建一个Project：  
ChildrenAlgorithm

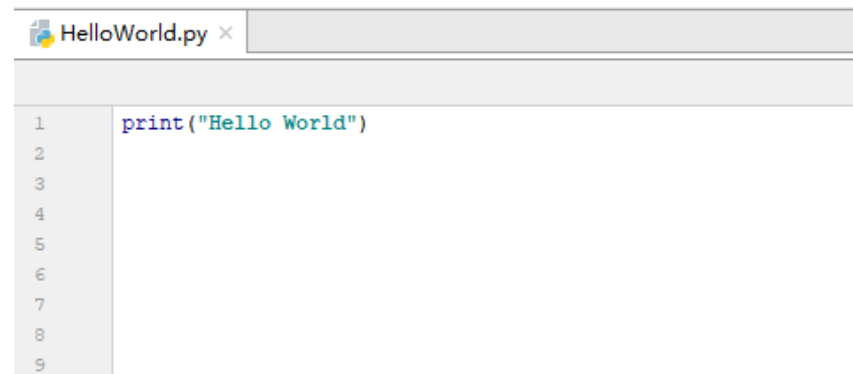
- File -> New... -> Python File

新建一个Python文件： HellowWorld.py

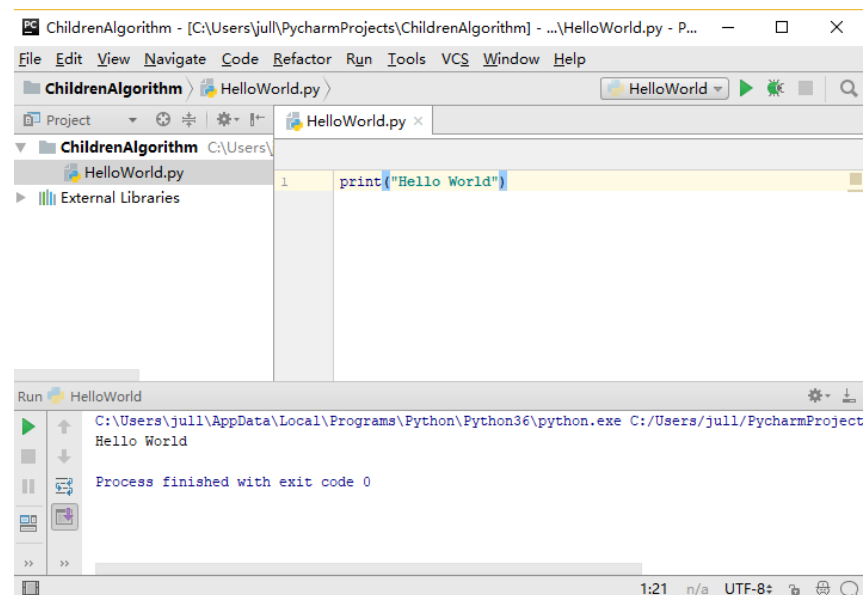


# 第一个Python程序 ——Hello World (2)

- 写代码
  - 运行程序
- “Run HelloWorld”



A screenshot of a code editor window titled 'HelloWorld.py'. The editor shows a single line of Python code: `print("Hello World")` on line 1. The line numbers 1 through 9 are visible on the left side of the editor.





# 编写运行Hello World的目的

- 证明程序的运行环境和工具可用
- 确认程序存储路径
- 设置好环境变量和IDE



**SAY 'HELLO WORLD'  
IN 28 DIFFERENT  
PROGRAMMING  
LANGUAGES**



# 编程第一步： 变量和常量



Variables



Constants

Coefficient

Variable

4x - 7 = 5

Operator

Constants

# 常量 (Constant)

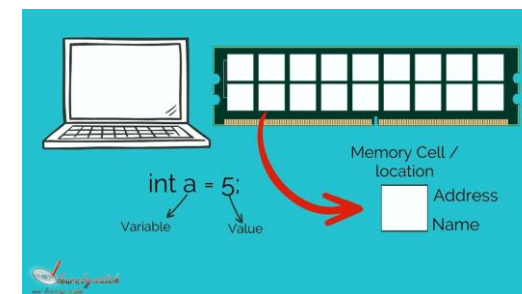
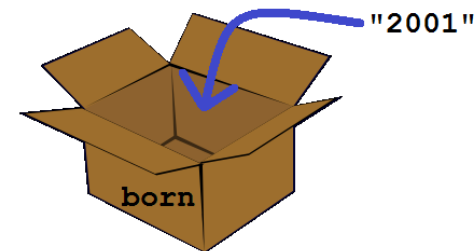
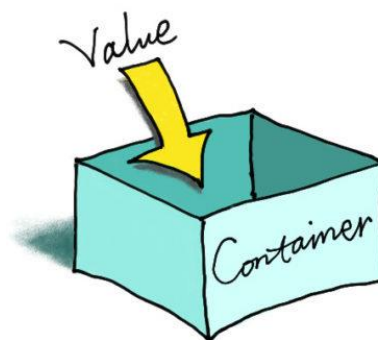
---

一个数、一个字.....一个确定的东西



# 变量 (Variable)

- 三要素
  - 名字
  - 类型
  - 值
- 对与同一个变量
  - 名字不能变
  - 值可以变
  - 类型看情况（和语言有关），在 Python 中不可变



## 赋值 (1-1)

- 把一个常量值赋给一个变量  
例如:  $a = 10$
- 从此这个变量有了一个值

$a = 10$



a = 10

b = a



a = 10

b = a

a = 5

a = ?   b = ?

## 赋值 (1-2)

- 把一个变量的值，赋给另一个变量  
例如：b = a
- 变量间的赋值就看当时
- 赋值后a的值改变，不会影响b的值



# 程序中的数组变量

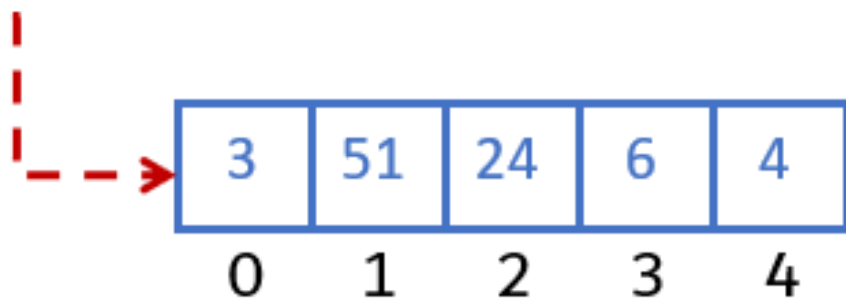
- 对应逻辑上的数据结构——数组（**NOTE:** 在后续的程序中，我们实际用到的是Python中的List而非Array，不过在逻辑上，我们用Python的List对应之前讲的“数组”概念）
- 变量名——数组的名字， e.g.: arr

```
SequentialSearch.py x
1 arr = [1,5,8,19,3,2,14,6,8,22,44,95,78]
2
3 tn = 95
```

```
SequentialSearch.py x
1 arr = [1,5,8,19,3,2,14,6,8,22,44,95,78]
变量名
3 tn = 95
```

# 数组的下标和元素值

arr



- 数组
  - 数组长度
  - 数组元素的下标
  - 数组元素的值
- 
- 数组元素的下标从0开始，依次递增，增量为1
  - 每个元素的下标不会变，但元素值可能会变

## 数组的赋值 (1)

- 把一个数据结构赋给一个变量

例如: `arr = [3, 51, 24, 6, 4]`

- 变量代表这个数据结构

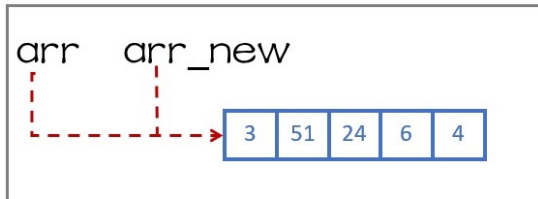
`arr = [3, 51, 24, 6, 4]`

`arr`

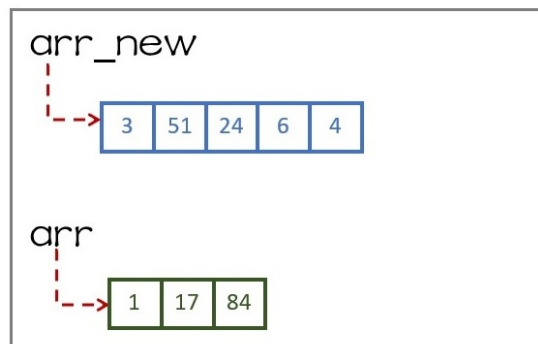


```
arr = [3, 51, 24, 6, 4]
```

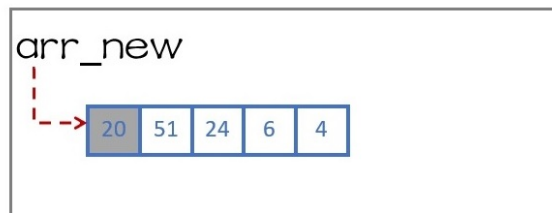
```
arr_new = arr
```



```
arr = [1, 17, 84]
```



```
arr_new[0] = 20
```



## 数组的赋值 (2)

- 把一个代表数据结构（例如数组）的变量赋给另一个变量
- 变量代表这个数组
- 不对这个变量进行新的赋值
  - 变量不会代表新的
- 数组中具体元素的值是可以改变的



# 练习题 (1)

- 1. 下面这段代码的输出内容是什么？

```
a = 10
b = a
a = 5
print(a, b)
```

- 2. 有一个数组arr，长度为10，所有元素值是一个递增的等差数列，相邻元素的差值为2，第一个元素值为1.

问： (1) 数组最后一个元素的下标是什么？

(2) arr[3]的值是什么？

(3) 将其中第三个和第五个元素的值交换，然后按顺序写出arr各个元素的取值

## 练习题 (2)

- 3. 一个数组arr, 长度为L, 已知  $1 < a < L - 1$

问: (1) 第一个元素和最后一个元素的下标分别是什么?

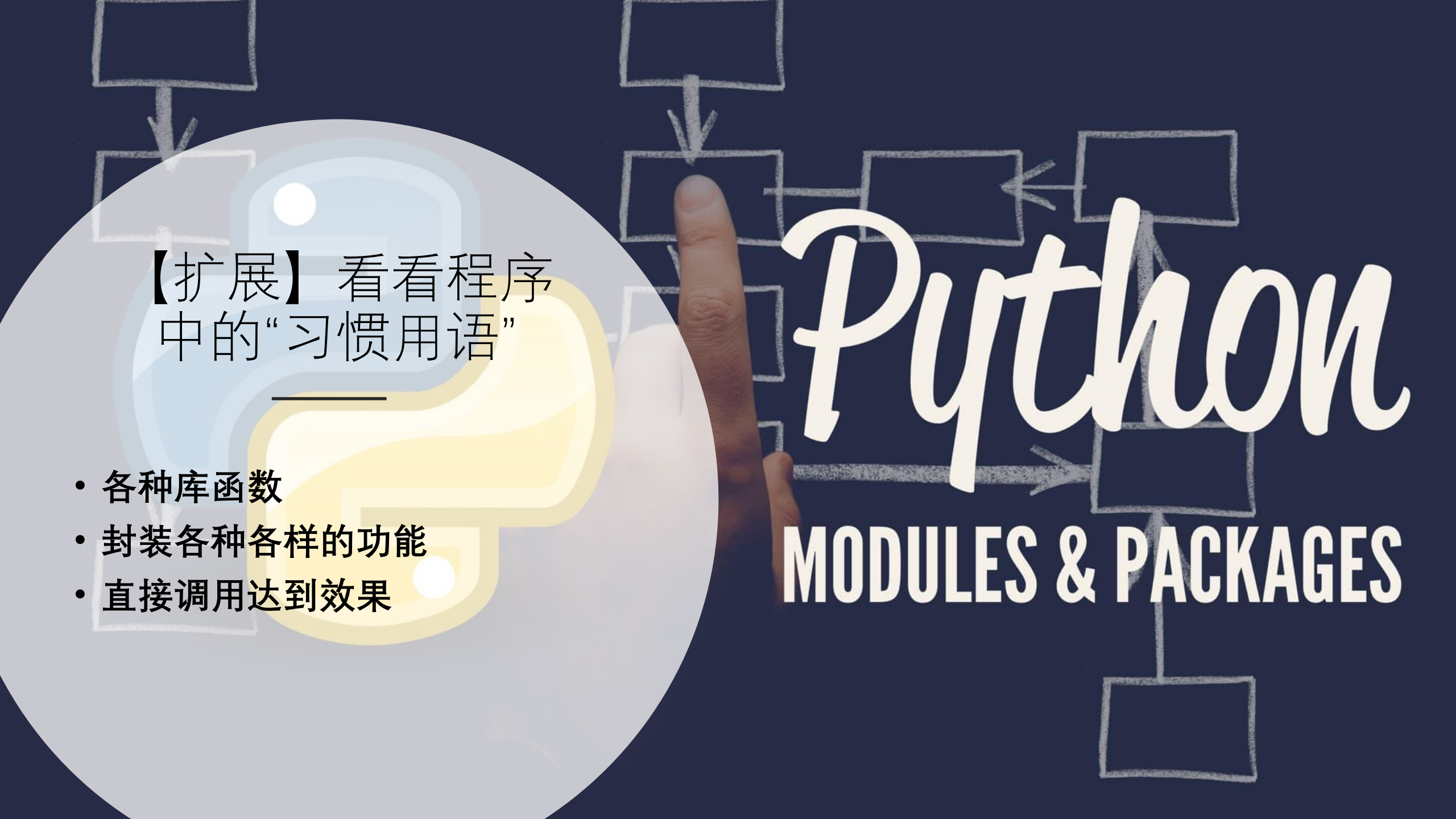
(2) 下标为a的元素前面那个元素的下标是什么? 后面那个元素是什么?

(3) 要把第a+1个元素的值赋为10, 应该怎么写代码?

- 4. 下面这段代码的输出内容是什么?

```
arr = [1, 2, 3]
arr_new = arr
arr = [4, 5, 6, 7]
arr_new[0] = 10
```

```
print(arr)
print(arr_new)
```

The background features a dark blue surface with faint, hand-drawn white flowcharts consisting of rectangular boxes and arrows. A large, semi-transparent circle is positioned on the left side, containing a stylized Python logo in light blue and yellow. A hand is visible, with a finger pointing towards the right side of the image.

## 【扩展】看看程序中的“习惯用语”

- 各种库函数
- 封装各种各样的功能
- 直接调用达到效果

# Python

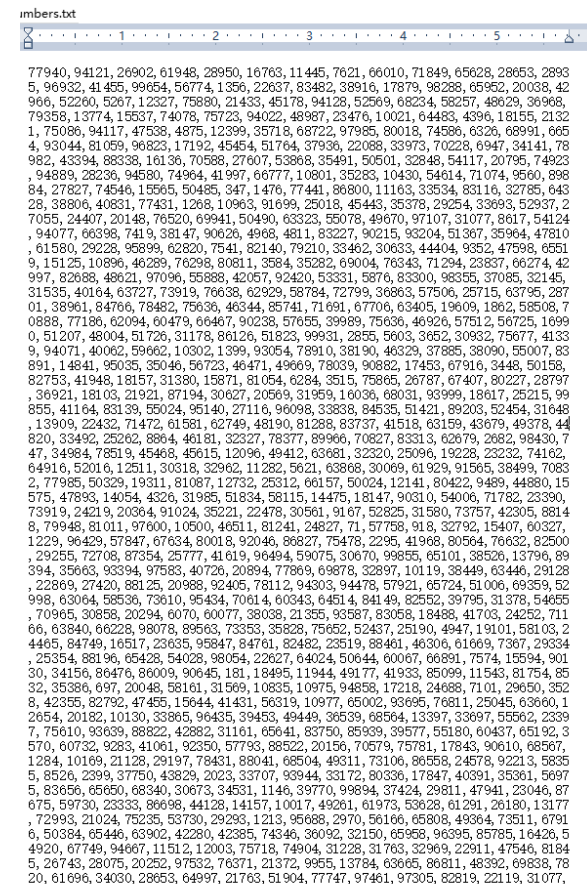
## MODULES & PACKAGES

# 从文件里面读取数字

- 文件中可以存储成百上千的数字
- 读取文件中数字的Python代码:

```
file = open("numbers.txt", "r")
arr = []
for val in file.read().split(','):
    arr.append(int(val))

file.close()
```



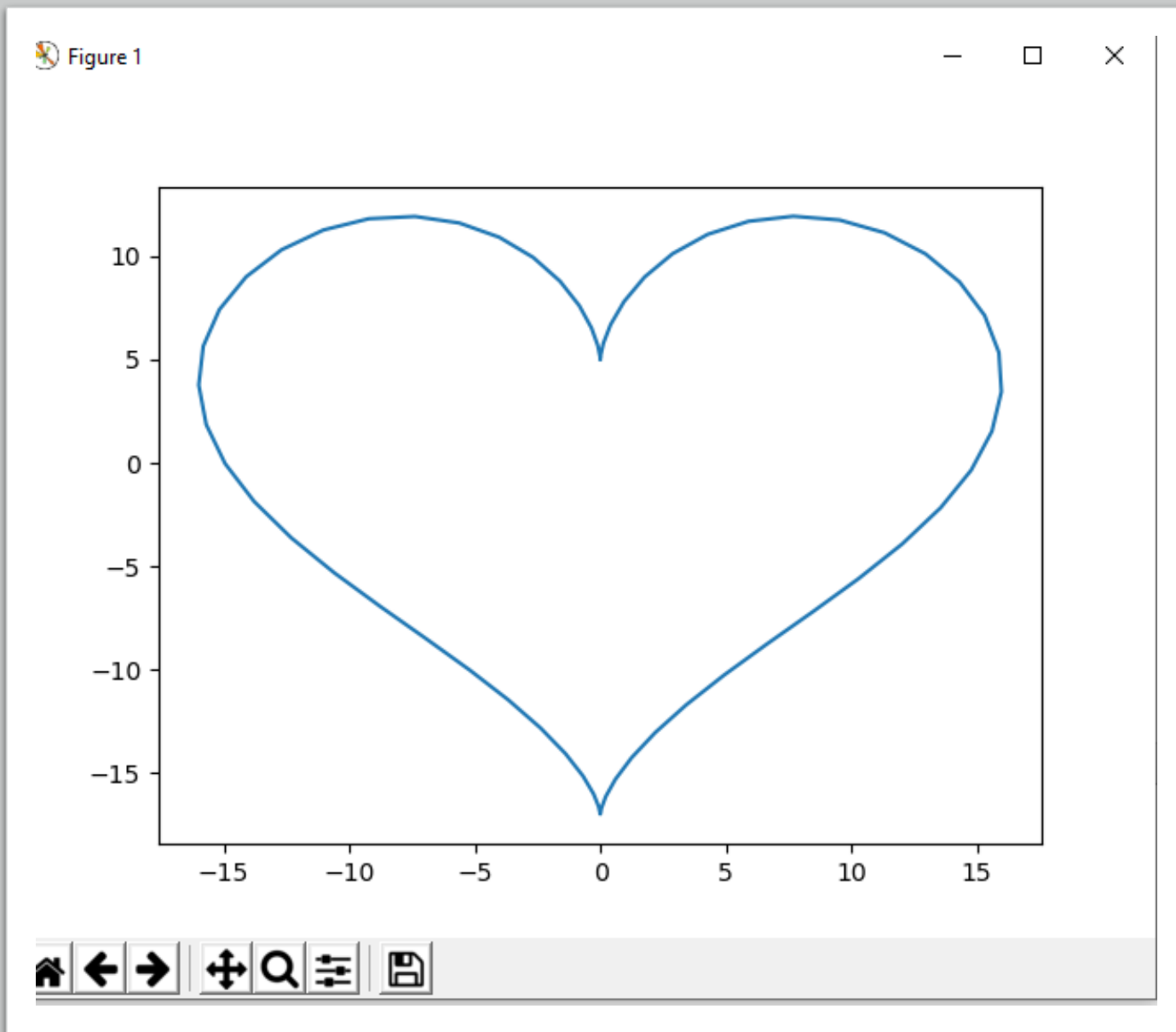
```
numbers.txt
77940, 94121, 26902, 61948, 28950, 16763, 11445, 7621, 66010, 71849, 65628, 28653, 2893
5, 96932, 41456, 99654, 56774, 1356, 22637, 83482, 38916, 17879, 98288, 65952, 20038, 42
966, 52260, 5267, 12327, 75880, 21433, 45178, 94128, 52569, 68234, 58257, 48629, 36968,
79358, 13774, 15537, 74078, 75723, 94022, 48987, 23476, 10021, 64483, 4396, 18155, 2132
1, 75086, 94117, 47538, 4875, 12399, 35718, 68722, 97985, 80018, 74586, 6326, 68991, 665
4, 93044, 81059, 96823, 17192, 45454, 51764, 37936, 22088, 33973, 70228, 6947, 34141, 78
982, 43394, 88338, 16136, 70588, 27607, 53868, 35491, 50501, 32848, 54117, 20795, 74923
, 94889, 28236, 94580, 74964, 41997, 66777, 10801, 35283, 10430, 54614, 71074, 9560, 898
84, 27827, 74546, 15565, 50485, 347, 1476, 77441, 86800, 11163, 33534, 83116, 32785, 643
28, 38806, 40631, 77431, 1268, 10963, 91639, 25018, 45443, 35378, 23254, 33093, 52937, 2
7055, 24407, 20148, 76520, 69941, 50490, 63323, 55078, 49670, 97107, 31077, 8617, 54124
, 94077, 66398, 7419, 38147, 90626, 4968, 4811, 83227, 90215, 93204, 51367, 35964, 47810
, 61580, 29228, 95899, 62820, 7541, 82140, 79210, 33462, 30633, 44404, 9352, 47598, 6551
9, 15125, 10896, 46289, 76298, 80811, 3584, 35282, 69004, 76343, 71294, 23837, 66274, 42
997, 82688, 48621, 97096, 55888, 42057, 92420, 53331, 5876, 83300, 98355, 37085, 32145,
31535, 40164, 63727, 73919, 76638, 62929, 58784, 72799, 36863, 57506, 25715, 63795, 287
01, 38961, 84766, 78482, 75636, 46344, 85741, 71691, 67706, 63405, 19609, 1862, 58508, 7
0888, 77186, 62094, 60479, 66467, 90238, 57655, 39989, 75636, 46926, 57512, 56725, 1699
0, 51207, 48004, 51726, 31178, 86126, 51823, 99931, 2855, 5603, 3652, 30932, 75677, 4133
9, 94071, 40062, 59662, 10302, 1399, 93054, 78910, 38190, 46329, 37885, 38090, 55007, 83
891, 14841, 95035, 35046, 56723, 46471, 49669, 78039, 90882, 17453, 67916, 3448, 50158,
82753, 41948, 18157, 31380, 15871, 81054, 6284, 3515, 75865, 26787, 87407, 80227, 28787
, 36921, 18103, 21921, 87194, 30627, 20569, 31959, 16036, 68031, 93999, 18617, 25215, 99
855, 41164, 83139, 55024, 95140, 27116, 96098, 33838, 84535, 51421, 89203, 52454, 31648
, 13909, 22432, 71472, 61581, 62749, 48190, 81288, 83737, 41518, 63159, 43679, 49378, 44
820, 33492, 25262, 8864, 46181, 32327, 78377, 89966, 70827, 83313, 62679, 2682, 98430, 7
47, 34984, 78519, 45468, 45615, 12096, 49412, 63681, 32320, 25096, 19228, 23232, 74162,
64916, 52016, 12511, 30318, 32962, 11282, 5621, 63866, 30069, 61929, 91565, 38499, 7083
2, 77985, 50329, 19311, 81087, 12732, 25312, 66157, 50024, 12141, 80422, 9489, 44880, 15
575, 47893, 14054, 4326, 31985, 51834, 58115, 14475, 18147, 90310, 54006, 71782, 23390,
73919, 24219, 20364, 91024, 35221, 22478, 30561, 9167, 52825, 31580, 73757, 42305, 8814
8, 79948, 81011, 97600, 10500, 46511, 81241, 24827, 71, 57758, 918, 32792, 15407, 60327,
1229, 96429, 57847, 67634, 80018, 92046, 86627, 75478, 2295, 41968, 80564, 76632, 82500
, 29255, 72708, 87354, 25777, 41619, 96494, 59075, 30670, 98555, 65101, 38526, 13796, 89
394, 35663, 93394, 97593, 40726, 20894, 77869, 69878, 32897, 10119, 38449, 63446, 29128
, 22869, 27420, 88125, 20988, 92405, 78112, 94303, 94478, 57921, 65724, 51006, 69359, 52
998, 63064, 58536, 73610, 95434, 70614, 60343, 64514, 84149, 82552, 39795, 31378, 54655
, 70965, 30858, 20294, 6070, 60077, 38038, 21355, 93587, 83058, 18488, 41703, 24252, 711
66, 63840, 66228, 98078, 89563, 73353, 35828, 75652, 52437, 25190, 4947, 19101, 58103, 2
4465, 84749, 16517, 23635, 95847, 84761, 82482, 23519, 88461, 46306, 61669, 7367, 29334
, 25354, 88196, 65428, 54028, 98054, 22627, 64024, 50644, 60067, 66891, 7574, 15594, 901
30, 34156, 86476, 86009, 90645, 181, 18495, 11944, 49177, 41933, 85099, 11543, 81754, 85
32, 35386, 697, 20048, 58161, 31569, 10835, 10975, 94858, 17218, 24688, 7101, 29650, 352
8, 42355, 82792, 47455, 15644, 41431, 56319, 10977, 65002, 93695, 76811, 25045, 63660, 1
2654, 20162, 10130, 33868, 96435, 39453, 49449, 36539, 68564, 13397, 33697, 55562, 2339
7, 75610, 36339, 88322, 42882, 31161, 65641, 63750, 85393, 39577, 55180, 60437, 65192, 3
570, 60732, 8283, 41061, 92350, 67793, 88522, 20156, 70579, 75761, 17843, 90610, 69567,
1284, 10169, 21128, 29197, 78431, 88041, 68504, 49311, 73106, 86558, 24578, 92213, 5835
5, 8526, 2399, 37750, 43829, 2023, 33707, 93944, 33172, 80336, 17847, 40391, 35361, 5697
5, 83656, 65650, 68340, 30673, 34531, 1146, 39770, 99894, 37424, 29811, 47941, 23046, 87
72993, 21024, 75235, 53730, 29293, 1213, 95688, 2970, 56166, 65808, 49364, 73511, 6791
6, 50384, 65446, 63902, 42280, 42385, 74346, 36092, 32150, 65958, 96395, 85785, 16426, 5
4920, 67749, 94667, 11512, 12003, 75718, 74904, 31228, 31763, 32969, 22911, 47546, 8184
5, 26743, 28075, 20252, 97532, 76371, 21372, 9955, 13784, 63665, 86811, 48392, 69838, 78
20, 61696, 34030, 28653, 64997, 21763, 51904, 77747, 97461, 97305, 82819, 22119, 31077,
```

# 画一条心形曲线

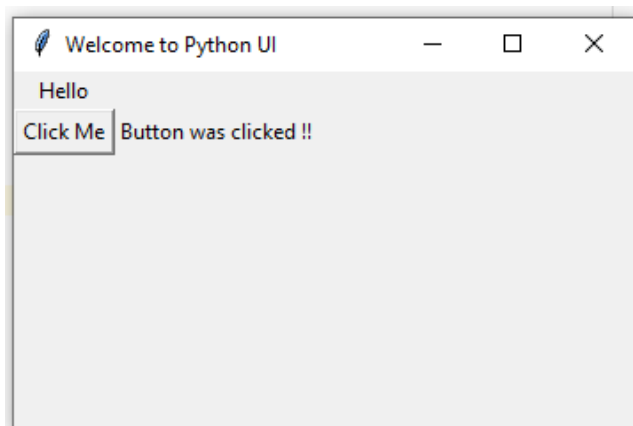
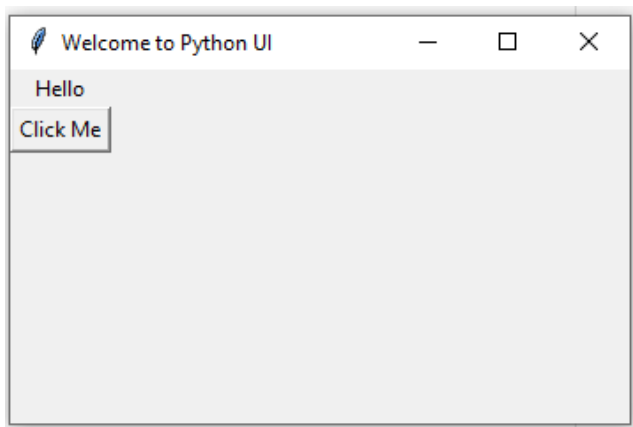
```
import matplotlib.pyplot as plt
import numpy as np
```

```
t = np.arange(0, 2*np.pi, 0.1)
x = 16*np.sin(t)**3
y = 13*np.cos(t) - 5*np.cos(2*t) \
    - 2*np.cos(3*t) - np.cos(4*t)
```

```
plt.plot(x, y)
plt.show()
```







# 制作用户界面

```
from tkinter import *
```

```
window = Tk()
```

```
window.title("Welcome to Python UI")  
window.geometry('350x200')
```

```
lb1 = Label(window, text="Hello")  
lb1.grid(column=0, row=0)
```

```
lb2 = Label(window, text="")  
lb2.grid(column=1, row = 1)
```

```
def clicked():  
    lb2.configure(text="Button was clicked !!")
```

```
btn = Button(window, text="Click Me", command=clicked)  
btn.grid(column=0, row=1)
```

```
window.mainloop()
```



谢谢！