```
C程序是运行前直接编译成CPU能执行的机器码，所以非常快。
```


```
//回车代表结束
```

```
//类的定义
class Student(object):
    pass
//类的实例化
bart = Student()
//
bart.name = 'tjs';
//由于类可以起到模板的作用，因此，可以在创建实例的时候，把一些我们认为必须绑定的属性强制填写进去。通过定义一个特殊的__init__方法，在创建实例的时候，就把name，score等属性绑上去：
//相当于java中的构造器
class Student(object):
    def __init__(self, name, score):
        self.name = name
        self.score = score

//它体现了 Python 中对象的“动态性”：Python 类的实例（对象）可以在运行时随时添加新的属性，而这些属性只属于这个对象自己，不会影响其他同类的对象。

//可以把属性的名称前加上两个下划线__，在Python中，实例的变量名如果以__开头，就变成了一个私有变量（private），只有内部可以访问，外部不能访问
            //本质是改名字了
            class Student(object):
    def __init__(self, name, score):
       self._name  = name  //private属性
        self._score = score

    def print_score(self):
        print('%s: %s' % (self.__name, self.__score))

//是以双下划线开头，并且以双下划线结尾的，是特殊变量，特殊变量是可以直接访问的，__name__
            //Python本身没有任何机制阻止你干坏事，一切全靠自觉。
       //当子类和父类都存在相同的run()方法时，我们说，子类的run()覆盖了父类的run()，在代码运行的时候，总是会调用子类的run()。这样，我们就获得了继承的另一个好处：多态



  //编译类型：确定有哪些变量和方法
```

```
    运行类型：确定具体的内容

        666  对于Python这样的动态语言来说，则不一定需要传入Animal类型。我们只需要保证传入的对象有一个run()方法就可以了：
```

获取对象的信息

```
通过type函数
    总是优先使用isinstance()判断类型，可以将指定类型及其子类“一网打尽”。
    如果要获得一个对象的所有属性和方法，可以使用dir()函数
    setattr(obj, 'y', 19) # 设置一个属性'y'
    >>> hasattr(obj, 'x') # 有属性'x'吗？
True
```

定义了一个类属性后，这个属性虽然归类所有，但类的所有实例都可以访问到

文件操作

```
1.f = open('文件地址','r'); //打开文件，读取模式
f.read();
2.    with open('input.txt') as f:
        print(f.read())
 //with自带f.close(); 如果出错她会自动调用，但是仍然需要 except FileNotFoundError 表示错误。
3.'rb' 打开二进制文件

4.with open('/Users/michael/test.txt', 'w') as f:
    f.write('Hello, world!')//打开文件，以及写入
    不用考虑关闭文件因为with自带
    //如果文件已存在，会直接覆盖（相当于删掉后新写入一个文件）
 5.传入'a'以追加（append）模式写入。
```

StringIO

在内存中读写文件 （模拟）

StringIO

先创建在再使用

```
from io import StringIo
f = StringIO()
f.write('hello')
f.write('')
print(f.getvalue())
输出：hello world!
//可以初始化
f = StringIO('Hello!\nHi!\nGoodbye!')
```

```
StringIO`操作的只能是`str`，如果要操作二进制数据，就需要使用`BytesIO
```

文件的编码

信息翻译成二进制

关键用对应的编码去翻译二进制（等会试试） UTF-8通用编码

文件的读取操作

```
open(name,mode,encoding) encoding编码格式
 f = open('python.txt','r',encoding = 'UTF-8') //encoding关键字传参。
    w 覆盖写入，不存在则创建
    a 追加 不存在则创建
  f.read(); //读取所有  f.read(5) 读取5个字节string
f.readlines()
 lines = f.readlines()   //封装到列表中
    print(f"f的作用{变量}")
    //文件不关闭，read（）方法从上次读取的位置读取。
  f.readline()//仅读取一行。

  //for循环读取对象
    for line in open("python.txt","r")
        print(line) //line每次循环读取一行
  f.close()  //一直占用，文件无法操作。
```

```
with open() as f:
    for line in f:
        print(line)
 with open() as //自动close（） 里面执行完之后
            split()

f.write("hello,word") 写入文件
            f.flush() 刷新文件
```

异常

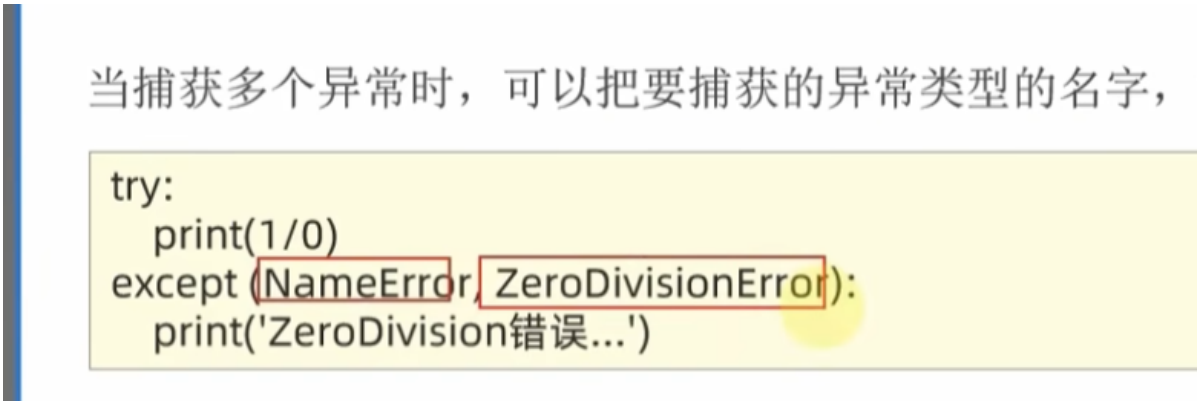
捕获异常

```
try:

except:
else:
     print("无异常时执行")
finally:
    print（'不过有无错误都执行'）
```

捕获指定的异常

```
try:
except NameError as e:
    print("出现了变量未定义的异常")
```



捕获多个异常

捕获所有异常

```
try:
except Exception as e:
print（“出现异常”）；
```

不处理就不断抛出

重新复习

```
字符串是以单引号'或双引号"
//"I\'m\"ok\" !"

#重点r'内部的字符串默认不转义'
print(r''\\\t\\');
//三个点表示多行
print('''line1
line2
line3''')
```

/ 除法计算结果是浮点数，即使是两个整数恰好整除，结果也是浮点数：

//底边除，向下取整，得到的是整数

字符串