
python
import turtle
turtle.setup(650,350,200,200)
turtle.penup()
turtle.fd(-250)
turtle.pendown()
turtle.pensize(25)
turtle.pencolor("purple")
turtle.seth(-40)
for 1 in range(4):
turtle.circle(40, 80)
turtle.circle(-40, 80)
turtle.fd(40)
turtle.circle(40, 80/2)
turtle.fd(40)
turtle.circle(16, 180)
turtle.fd(40 * 2/3)

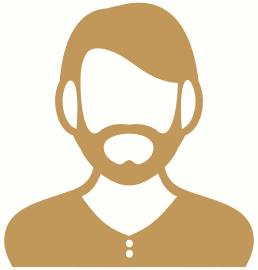
Python语言程序设计

Python程序设计思维

嵩 天
北京理工大学

单元开篇

Python程序设计思维



- 计算思维与程序设计
- 计算生态与Python语言
- 用户体验与软件产品
- 基本的程序设计模式

python
语言程序设计

计算思维与程序设计

计算思维

第3种人类思维特征

- 逻辑思维：推理和演绎，数学为代表，A->B B->C A->C
- 实证思维：实验和验证，物理为代表，引力波<-实验
- 计算思维：设计和构造，计算机为代表，汉诺塔递归

计算思维

抽象和自动化

- 计算思维：Computational Thinking
- 抽象问题的计算过程，利用计算机自动化求解
- 计算思维是基于计算机的思维方式

计算思维

计数求和：计算1-100的计数和

$$S = \frac{(a_1 + a_n)n}{2}$$

```
s = 0
for i in range(1, 101):
    s += i
```

逻辑思维

数学家高斯的玩儿法

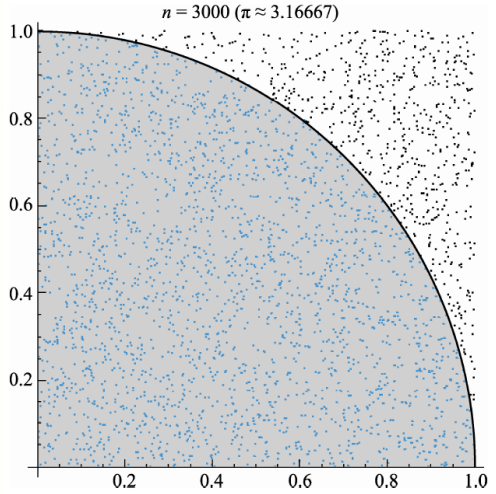
计算思维

现代人的新玩儿法

计算思维

圆周率的计算

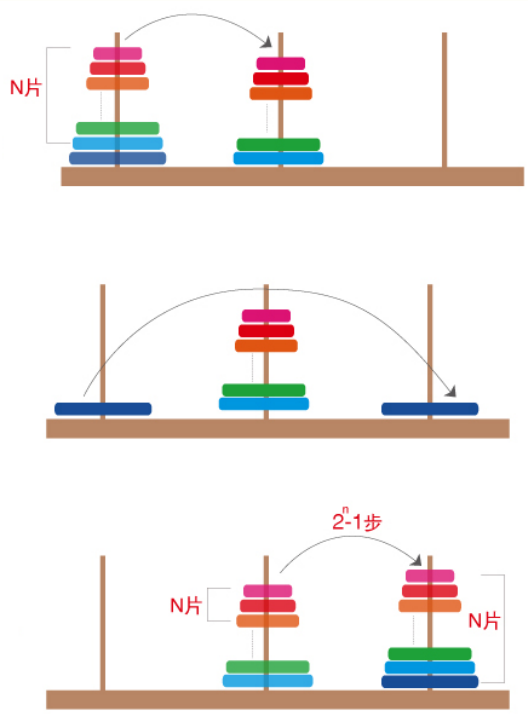
$$\pi = \sum_{k=0}^{\infty} [\frac{1}{16^k}\left(\frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6}\right)]$$



逻辑思维

计算思维

计算思维



汉诺塔问题

```
count = 0
def hanoi(n, src, dst, mid):
    … (略)
hanoi(3, "A", "C", "B")
print(count)
```

```
>>>
1:A->C
2:A->B
1:C->B
3:A->C
1:B->A
2:B->C
1:A->C
7
```

逻辑思维 $2^n - 1$ 计算思维

计算思维

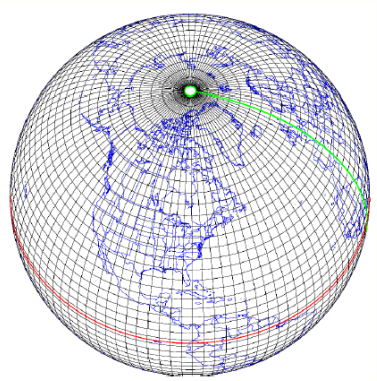
经验

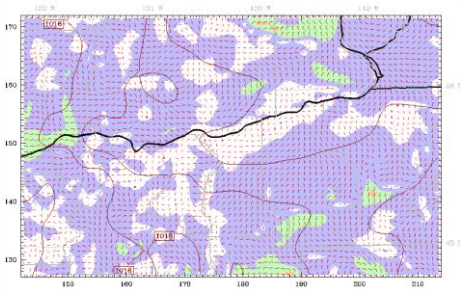
猜

天气预报

MM5模型

@超算



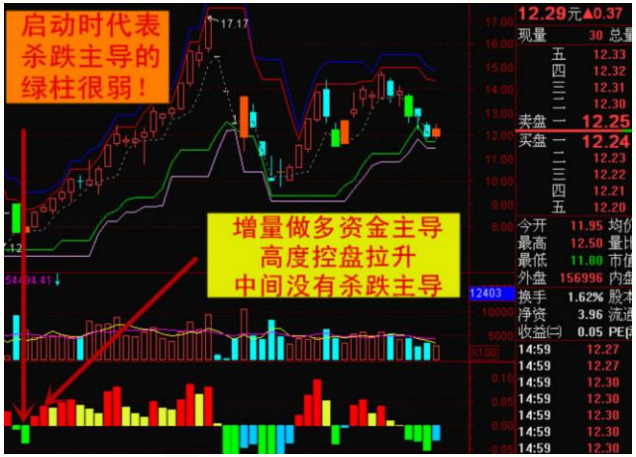


实证思维+逻辑思维

计算思维

计算思维

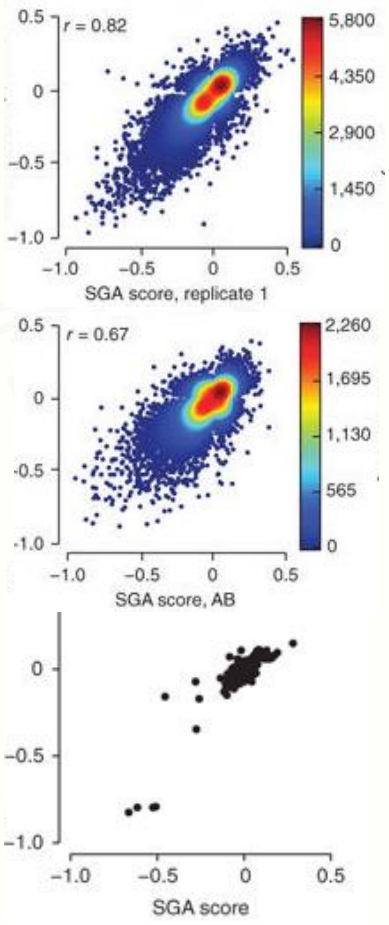
量化分析



猜

机器学习

自动交易



实证思维+逻辑思维

计算思维

计算思维

抽象问题的计算过程，利用计算机自动化求解

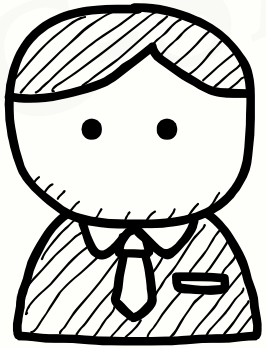
- **计算思维基于计算机强大的算力及海量数据**
- **抽象计算过程，关注设计和构造，而非因果**
- **以计算机程序设计为实现的主要手段**

计算思维与程序设计

编程是将计算思维变成现实的手段

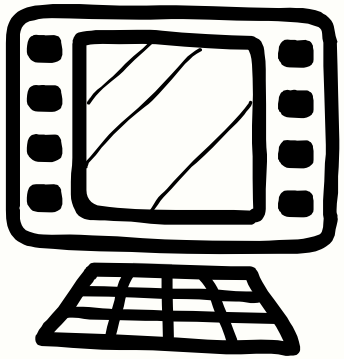
抽象

设计和构造

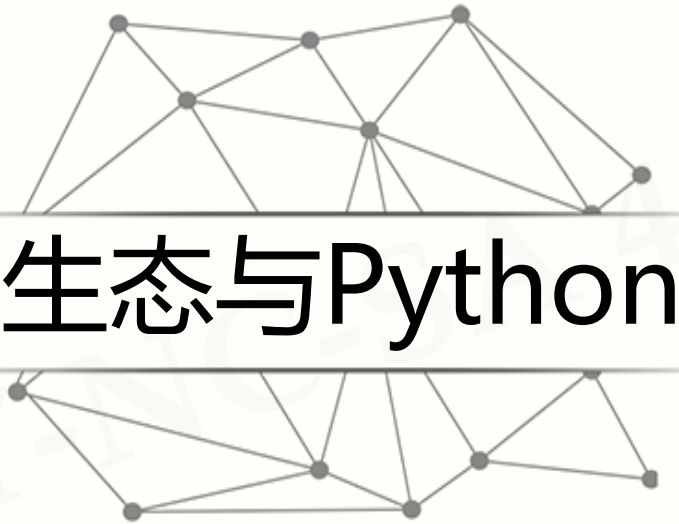


自动化

编程



计算思维 真的很有用...



计算生态与Python语言

计算生态

从开源运动说起...

- 1983, Richard Stallman启动GNU项目
- 1989, GNU通用许可协议诞生

自由软件时代到来

计算生态

从开源运动说起...

- 1991, Linus Torvalds发布了Linux内核
- 1998, 网景浏览器开源，产生了Mozilla

开源生态逐步建立

计算生态

从开源运动说起...

V.S.

- 1983, Richard Stallman

大教堂模式

- 1991, Linus Torvalds

集市模式

计算生态

开源思想深入演化和发展，形成了计算生态

计算生态以开源项目为组织形式，充分利用“共识原则”和“社会利他”组织人员，在竞争发展、相互依存和迅速更迭中完成信息技术的更新换代，形成了技术的自我演化路径。

计算生态

没有顶层设计、以功能为单位、具备三个特点

- **竞争发展**
- **相互依存**
- **迅速更迭**

计算生态与Python语言

- 以开源项目为代表的大量第三方库

  Python语言提供 >13万个第三方库

- 库的建设经过野蛮生长和自然选择

  同一个功能，Python语言2个以上第三方库

计算生态与Python语言

- 库之间相互关联使用，依存发展

  Python库间广泛联系，逐级封装

- 社区庞大，新技术更迭迅速

  AlphaGo深度学习算法采用Python语言开源

计算生态与Python语言

API != 生态

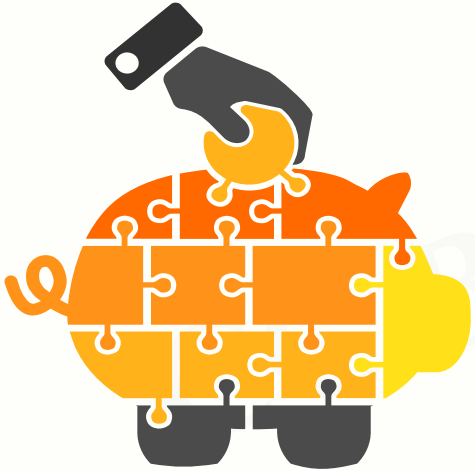
计算生态的价值

创新：跟随创新、集成创新、原始创新

- **加速科技类应用创新的重要支撑**
- **发展科技产品商业价值的重要模式**
- **国家科技体系安全和稳固的基础**

计算生态的运用

刀耕火种 -> 站在巨人的肩膀上



- **编程的起点不是算法而是系统**
- **编程如同搭积木，利用计算生态为主要模式**
- **编程的目标是快速解决问题**

计算生态

优质的计算生态

理解和运用计算生态

用户体验与及软件产品

用户体验

实现功能 -> 关注体验

- **用户体验指用户对产品建立的主观感受和认识**
- **关心功能实现，更要关心用户体验，才能做出好产品**
- **编程只是手段，不是目的，程序最终为人类服务**

提高用户体验的方法

方法1：进度展示

- 如果程序需要计算时间，可能产生等待，请增加进度展示
- 如果程序有若干步骤，需要提示用户，请增加进度展示
- 如果程序可能存在大量次数的循环，请增加进度展示

提高用户体验的方法

方法2：异常处理

- 当获得用户输入，对合规性需要检查，需要异常处理
- 当读写文件时，对结果进行判断，需要异常处理
- 当进行输入输出时，对运算结果进行判断，需要异常处理

提高用户体验的方法

其他类方法

- 打印输出：特定位置，输出程序运行的过程信息
- 日志文件：对程序异常及用户使用进行定期记录
- 帮助信息：给用户多种方式提供帮助信息

软件程序 -> 软件产品

用户体验是程序到产品的关键环节

基本的程序设计模式

基本的程序设计模式

从IPO开始...

- I：Input 输入，程序的输入
- P：Process 处理，程序的主要逻辑
- O：Output 输出，程序的输出

基本的程序设计模式

从IPO开始...

- 确定IPO：明确计算部分及功能边界
- 编写程序：将计算求解的设计变成现实
- 调试程序：确保程序按照正确逻辑能够正确运行

基本的程序设计模式

自顶向下设计

- I：Input 输入，程序的输入
- P：Process 处理，程序的主要逻辑
- O：Output 输出，程序的输出

基本的程序设计模式

自顶向下设计

- I：Input 输入，程序的输入
- P：Process 处理，程序的主要逻辑
- O：Output 输出，程序的输出

基本的程序设计模式

模块化设计

- 通过函数或对象封装将程序划分为模块及模块间的表达
- 具体包括：主程序、子程序和子程序间关系
- 分而治之：一种分而治之、分层抽象、体系化的设计思想

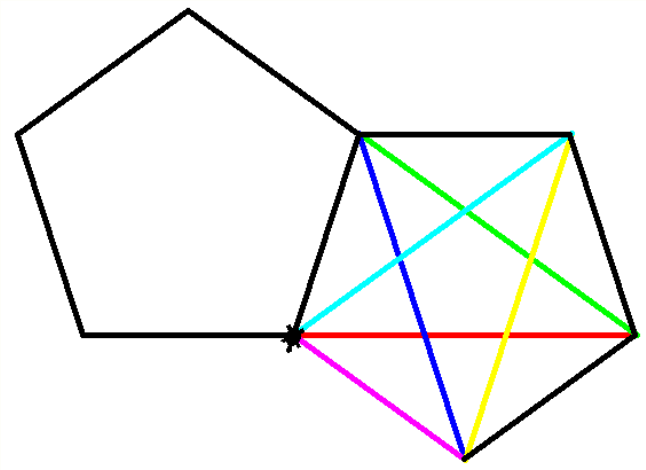
基本的程序设计模式

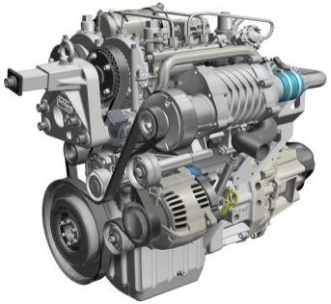
模块化设计

- 紧耦合：两个部分之间交流很多，无法独立存在
- 松耦合：两个部分之间交流较少，可以独立存在
- 模块内部紧耦合、模块之间松耦合

基本的程序设计模式

配置化设计





程序引擎

配置文件

基本的程序设计模式

配置化设计

- **引擎+配置**：程序执行和配置分离，将可选参数配置化
- 将程序开发变成配置文件编写，扩展功能而不修改程序
- 关键在于接口设计，清晰明了、灵活可扩展

应用开发的四个步骤

从应用需求到软件产品

- 1 产品定义
- 3 设计与实现

- 2 系统架构
- 4 用户体验

应用开发的四个步骤

从应用需求到软件产品

- 1 **产品定义**：对应用需求充分理解和明确定义

  产品定义，而不仅是功能定义，要考虑商业模式

- 2 **系统架构**：以系统方式思考产品的技术实现

  系统架构，关注数据流、模块化、体系架构

应用开发的四个步骤

从应用需求到软件产品

- 3 **设计与实现**：结合架构完成关键设计及系统实现

  结合可扩展性、灵活性等进行设计优化

- 4 **用户体验**：从用户角度思考应用效果

  用户至上，体验优先，以用户为中心

单元小结

Python程序设计思维

- **计算思维：抽象计算过程和自动化执行**
- **计算生态：竞争发展、相互依存、快速更迭**
- **用户体验：进度展示、异常处理等**
- **IPO、自顶向下、模块化、配置化、应用开发的四个步骤**