**AI**

## 人工智能

**1 人工智能（Artificial Intelligence），英文缩写为AI。**它是研究、开发用于模拟、延伸和扩展人的智能的理论、方法、技术及应用系统的一门新的技术科学。

**2 人工智能是计算机科学的一个分支，它企图了解智能的实质**，并生产出一种新的能以人类智能相似的方式做出反应的智能机器，该领域的研究包括机器人、语言识别、图像识别、自然语言处理和专家系统等。人工智能从诞生以来，理论和技术日益成熟，应用领域也不断扩大，可以设想，未来人工智能带来的科技产品，将会是人类智慧的容器”。人工智能可以对人的意识、思维的信息过程的模拟。人工智能不是人的智能，但能像人那样思考、也可能超过人的智能。

**3 人工智能是一门极富挑战性的科学**，从事这项工作的人必须懂得计算机知识，心理学和哲学。人工智能是包括十分广泛的科学，它由不同的领域组成，如机器学习，计算机视觉等等，总的说来，人工智能研究的一个主要目标是使机器能够胜任一些通常需要人类智能才能完成的复杂工作。但不同的时代、不同的人对这种复杂工作”的理解是不同的。 [1] 2017年12月，人工智能入选2017年度中国媒体十大流行语”

**4 人工智能的定义可以分为两部分，即人工”和智能”。**

人工”比较好理解，争议性也不大。有时我们会要考虑什么是人力所能及制造的，或者人自身的智能程度有没有高到可以创造人工智能的地步，等等。但总的来说，人工系统”就是通常意义下的人工系统。

**人工智能AI（Artificial Intelligence）是在计算机科学、控制论、信息论、神经心理学、哲学、语言学等多种学科研究的基础上发展起来的一门交叉学科。**

**AI与空间技术、原子技术一起誉为21世纪的三大科学技术成就，前三次工业革命主要是延长了人手的功能，AI延伸了人脑的功能，实现了脑力劳动的自动化。**

关于什么是智能”，就问题多多了。这涉及到其它诸如意识（CONSCIOUSNESS）、自我（SELF）、思维（MIND）（包括无意识的思维（UNCONSCIOUS\_MIND））等等问题。人唯一了解的智能是人本身的智能，这是普遍认同的观点。但是我们对我们自身智能的理解都非常有限，对构成人的智能的必要元素也了解有限，所以就很难定义什么是人工”制造的智能”了。因此人工智能的研究往往涉及对人的智能本身的研究。其它关于动物或其它人造系统的智能也普遍被认为是人工智能相关的研究课题。

人工智能在计算机领域内，得到了愈加广泛的重视。并在机器人，经济政治决策，控制系统，仿真系统中得到应用。尼尔逊教授对人工智能下了这样一个定义：人工智能是关于知识的学科――怎样表示知识以及怎样获得知识并使用知识的科学。”而另一个美国麻省理工学院的温斯顿教授认为：人工智能就是研究如何使计算机去做过去只有人才能做的智能工作。”这些说法反映了人工智能学科的基本思想和基本内容。即人工智能是研究人类智能活动的规律，构造具有一定智能的人工系统，研究如何让计算机去完成以往需要人的智力才能胜任的工作，也就是研究如何应用计算机的软硬件来模拟人类某些智能行为的基本理论、方法和技术。

人工智能是计算机学科的一个分支，二十世纪七十年代以来被称为世界三大尖端技术之一（空间技术、能源技术、人工智能）。也被认为是二十一世纪三大尖端技术（基因工程、纳米科学、人工智能）之一。这是因为近三十年来它获得了迅速的发展，在很多学科领域都获得了广泛应用，并取得了丰硕的成果，人工智能已逐步成为一个独立的分支，无论在理论和实践上都已自成一个系统。

人工智能是研究使计算机来模拟人的某些思维过程和智能行为（如学习、推理、思考、规划等）的学科，主要包括计算机实现智能的原理、制造类似于人脑智能的计算机，使计算机能实现更高层次的应用。人工智能将涉及到计算机科学、心理学、哲学和语言学等学科。可以说几乎是自然科学和社会科学的所有学科，其范围已远远超出了计算机科学的范畴，人工智能与思维科学的关系是实践和理论的关系，人工智能是处于思维科学的技术应用层次，是它的一个应用分支。从思维观点看，人工智能不仅限于逻辑思维，要考虑形象思维、灵感思维才能促进人工智能的突破性的发展，数学常被认为是多种学科的基础科学，数学也进入语言、思维领域，人工智能学科也必须借用数学工具，数学不仅在标准逻辑、模糊数学等范围发挥作用，数学进入人工智能学科，它们将互相促进而更快地发展。

**5 什么是智能？智能的本质是什么？**

根据对人脑已有的认识，结合智能的外在表现，从不同的角度、不同的侧面、用不同的方法对智能进行研究，提出了几种不同的观点，其中影响较大的观点有思维理论、知识阈值理论及进化理论。

**（1）思维理论：**

认为智能的核心是思维，人的一切智能都来自大脑的思维活动，人类的一切知识都是人类思维的产物，因而通过对思维规律与方法的研究可望揭示智能的本质。

**（2）知识阈值理论：**

认为智能行为取决于知识的数量及其一般化的程度，它把智能定义为：智能就是在巨大的搜索空间中迅速找到一个满意解的能力。

**（3）进化理论：**

人的本质能力是在动态环境中的行走能力、对外界事物的感知能力、维持生命和繁衍生息的能力，智能取决于感知和行为，因而智能是在系统与周围环境不断“刺激-反应”的交互发展和进化的。

该理论的核心是用控制取代表示，从而取消概念、模型及显式表示的知识，否定抽象对于智能及智能模拟的必要性，强调分层结构对于智能进化的可能性与必要性。

**（4）综合上述观点，可认为：智能是知识与智力的总和。其中，知识是一切智能行为的基础，而智力是获取知识并应用知识求解问题的能力。**

**6 智能的基本特征**

（1）感知能力：

通过视觉、听觉、触觉、嗅觉等感觉器官感知外部世界的能力。

80%以上信息通过视觉得到，10%信息通过听觉得到。

（2）记忆与思维能力

存储由感知器官感知到的外部信息以及由思维所产生的知识

对记忆的信息进行处理

（3）逻辑思维（抽象思维）

依靠逻辑进行思维。

思维过程是串行的。

容易形式化。

思维过程具有严密性、可靠性

（4）形象思维（直感思维）

依据直觉。

思维过程是并行协同式的。

形式化困难。

在信息变形或缺少的情况下仍有可能得到比较满意的结果

（5）顿悟思维（灵感思维）

不定期的突发性。

非线性的独创性及模糊性。

穿插于形象思维与逻辑思维之中

（6）学习能力

学习既可能是自觉的、有意识的，也可能是不自觉的、无意识的；既可以是有教师指导的，也可以是通过自己实践的。

（7）行为能力（表达能力）

人们的感知能力：用于信息的输入。

行为能力：信息的输出。

**7 人工智能本质**

**所谓人工智能就是用人工的方法在机器(计算机)上实现的智能，或者说是人们使机器具有类似于人的智能。由于人工智能是在机器上实现的，因此，又称为机器智能(Machine Intelligence)**

**通俗地说，人工智能就是要研究如何使机器具有能听、会说、能看、会写、能思维、会学习、能适应环境变化、能解决各种面临的实际问题等功能的一门学科。**

**8 图灵试验**

问： 请给我写出有关“第四号桥”主题的十四行诗。

答：不要问我这道题，我从来不会写诗。

问：34957加70764等于多少？

答：（停30秒后）105721

问：你会下国际象棋吗？

答：是的。

问：我在我的K1处有棋子K；你仅在K6处有棋子K，在R1处有棋子R。现在轮到你走，你应该下那步棋？

答：（停15秒钟后）棋子R走到R8处，将军！

图灵指出：“如果机器在某些现实的条件下，能够非常好地模仿人回答问题，以至提问者在相当长时间里误认它不是机器，那么机器就可以被认为是能够思维的。”

从表面上看，要使机器回答按一定范围提出的问题似乎没有什么困难，可以通过编制特殊的程序来实现。然而，如果提问者并不遵循常规标准，编制回答的程序是极其困难的事情。例如，提问与回答呈现出下列状况：

问：你会下国际象棋吗？

答：是的。

问：你会下国际象棋吗？

答：是的。

问：请再次回答，你会下国际象棋吗？

答：是的。

**第一种可明显地感到回答者是从知识库里提取简单的答案**

问：你会下国际象棋吗？

答：是的。

问：你会下国际象棋吗？

答：是的，我不是已经说过了吗？

问：请再次回答，你会下国际象棋吗？

答：你烦不烦，干嘛老提同样的问题。

**第二种则具有分析综合的能力，回答者知道观察者在反复提出同样的问题**

**9 AI基础**

**（1）哲学（公元前428年至今）**

哲学家们标志出了AI的大部分思想，但实现成为一门规范科学的飞跃就要求在三个基础领域完成一定程度的数学形式化：逻辑、计算和概率。

**（2）数学（约800年至今）**

概率起源于对赌博问题可能结果的描述，成为所有需要定量的科学的无价之宝，帮助对付不确定的测量和不完备的理论。（Cardano，16世纪）Bayes提出了根据新证据更新概率的法则。（18世纪）Bayes分析形成了大多数AI系统中不确定推理的现代方法的基础。

**（3）经济学（1776年至今）**

我们如何决策以获得最大效益？Adam Smith是第一个把经济学当作科学来看待，认为经济是个人代理之间的协调过程，这些代理追求自己的经济利益的最大化。对于“偏好的结果”（效用）的数学处理，由Walras完成形式化。决策理论把概率和效用结合起来，为在不确定条件下进行决策提供了形式化和完整的框架。（适用于“宏观”经济）“微观”经济则更像博奕游戏在他人不合作情况下如何获得最大效益？

**（4）神经科学（1861年至今）**

大脑是如何处理信息的

**（5）心理学（1879年至今）**

研究方法：内省 vs. 心理测量

行为主义者只研究对动物的感知和它所引发的行动的度量，排斥精神的结构，包括知识、信念、目标和推理步骤。认知心理学把大脑当作信息处理装置。基于知识的智能体的三个步骤：

把刺激翻译成内部表示，表示经过认知过程处理→新的表示，表示被翻译回到行动计算机模型的发展导致认知科学的创建。**普遍的观点：“认知理论就应该像计算机程序**

**（6）计算机工程（1940年至今）**

AI需要智能和人工制品，即计算机。

AI对主流计算机科学的影响

分时技术

交互式翻译器

使用窗口和鼠标的个人计算机

面向对象的编程

**（7）控制论（1948年至今）**

现代控制论和AI的共同点 ：设计出能随时间变化使目标函数最大化的系统。

控制论的主要工具是微积分和线性代数其主要研究对象是用固定的连续变量集描述的（线性）系统。AI的部分起因是寻求摆脱控制论的数学方法局限性的途径

不同的工具：逻辑推理和计算

不同的问题：语言、视觉、规划…

**（8）语言学（1957年至今）**

现代语言学的诞生： Chomsky(乔姆斯基)理论形式化，可以编程实现。知识表示的许多早期工作和语言紧密联系

**（9）总结**

**哲学：标出了AI的大部分重要思想**

**数学：使AI成为一门规范科学**

**经济学：决策理论**

**神经科学：网络，并行处理…**

**心理学：认知理论**

**计算机工程：AI的“载体”**

**控制论：反馈的思想**

**语言学：知识表示、语法**

**10 AI的发展**

**70年代**

前期以自然语言理解、知识表示研究为主

Winograd发表自然语言理解系统SHRDLU

Colmerauer创建PROLOG语言

Schank提出概念从属理论

Minskey提出框架知识表示法

Feigenbaum提出知识工程

**80年代**

专家系统广泛应用，出现了专家系统开发工具，开始兴起人工智能产业。

日本提出五代机计划

中国提出863计划（863-305）

**90年代**

人工神经网络的复兴

基于知识的系统 CYC

Deep Blue (1997.5.11)

分布式人工智能与多Agent系统

Robots，Softbot，集成自治系统

知识科学

**11 AI的研究目标**

（1）近期目标

使现有的电子数字计算机更聪明、更有用，使它不仅能做一般的数值计算及非数值信息的数据处理，而且能运用知识处理问题，能模拟人类的部分智能行为。建造智能机器人代替人类的部分智力劳动。

（2）远期目标

用自动机模仿人类的思维过程和智能行为。

（3）最终目标

机器智能实现生物智能的各项功能。

**12 AI研究的基本内容**

**（1）知识表示**

将人类知识形式化或者模型化。实际上就是对知识的一种描述，或者说是一组约定，一种计算机可以接受的用于描述知识的数据结构。分两大类：符号表示法，连接机制表示法。

**（2）机器感知**

使机器具有类似于人的感知能力，其中以机器视觉与机器听觉为主。机器视觉是让机器能够识别并理解文字、图像、物景等；机器听觉是让机器能识别并理解语言、声响等。模式识别与自然语言理解

**（3）机器思维**

对通过感知得来的外部信息及机器内部的各种工作信息进行有目的的处理，是AI研究中最重要、最关键的部分

**（4）机器学习**

研究如何是计算机具有类似人的学习能力，使它能通过学习自动地获取知识。计算机可以直接向书本学习，通过与人谈话和对环境的观察学习，并在实践中实现自我完善。机器学习研究的目标有3个：人类学习过程的认知模型；通用学习算法；构造面向任务的专用学习系统的方法。

**（5）机器行为**

与人的行为能力相对应，机器行为主要是指计算机的表达能力，即“说”、“写”、“画”等能力。对于智能机器人，它还应具有人的四肢功能，即能走路、能取物、能操作等。

**13 AI的主要研究领域**

**（1）自动定理证明**

数学领域中对臆测的定理寻求一个证明，一直被认为是一项需要智能才能完成的任务。证明定理时，不仅需要有根据假设进行演绎的能力，而且需要有某些直觉的技巧。例如数学家在求证一个定理时，会熟练地运用他丰富的专业知识，猜测应当先证明哪一个引理，精确判断出已有的哪些定理将起作用，并把主问题分解为若干子问题，分别独立进行求解。因此人工智能研究中机器定理证明很早就受到注视，并取得不少成果。定理证明的研究在人工智能方法的发展中曾起过重要的作用，例如使用谓词逻辑语言，其演绎过程的形式体系研究，帮助人们更清楚地理解推理过程的各个组成部分。许多其他领域的问题，也可以转化为定理证明问题，因此机器定理证明的研究具有普遍意义。

**（2）博弈**

博弈被认为是智能的活动，人工智能中主要是研究下棋程序，在六十年代就出现了很有名的西洋跳棋和国际象棋的程序，并达到了大师的水平。进入90年代，IBM公司以其雄厚硬件基础，支持开发后来被称之为“深蓝”的国际象棋系统，并为此开发了专用的芯片，以提高计算机的搜索速度。1996年2月，与国际象棋世界冠军卡斯帕罗夫进行了第一次比赛，经过六个回合的比赛之后，“深蓝”以2：4告负。1997年5月，系统经过改进以后，"深蓝"又第二次与卡斯帕罗夫交锋，并最终以3.5：2.5战胜了卡斯帕罗夫，在世界范围内引起了轰动。

2007年台北国际发明暨技术交易展览上，第三代智能机器人DOC 现场表演下棋。人工智能研究博弈的目的并不是为了让计算机与人进行下棋、打牌之类的游戏，而是通过对博弈的研究来检验某些人工智能技术是否能实现对人类智慧的模拟，促进人工智能技术深入一步的研究。

**（3）模式识别**

机器感知是机器智能的一个重要方面，是机器获取外部信息的基本途径。模式识别(pattern recognition)就是研究如何使机器具有感知能力的一个研究领域，其中主要研究对视觉模式及听觉模式的识别。模式是对一个物体或者摹写其他感兴趣实体定量或者结构的描述，而模式类是指具有某些共同属性的模式集合。用机器进行模式识别的主要内容是研究一种自动技术，依靠这种技术，机器可以自动或者尽可能少需要人工干预地把模式分配到它们各自的模式类中去。

**（4）机器视觉**

机器视觉(machine vision)或计算机视觉(computer vision)是用机器代替人眼睛进行测量或判断，是模式识别研究的一个重要方面。低层视觉主要执行预处理功能，如边缘检测、动目标检测、纹理分析、立体造型以及曲面色彩等，主要目的是使看见的对象更突出。这是还不是理解阶段。高层视觉主要是理解对象，需要掌握与对象相关的知识。国外，机器视觉的应用相当普及，主要集中在半导体及电子、汽车、冶金、食品饮料、零配件装配及制造行业。

国内机器视觉产品刚刚起步，目前主要集中在制药、印刷、包装、食品饮料等行业。

**（5）自然语言理解**

自然语言是人类之间信息交流的主要媒介，由于人类有很强的理解自然语言的能力，因此互相间的信息交流显得轻松自如。然而目前计算机系统和人类之间的交互几乎还只能使用严格限制的各种非自然语言(如C、Java等)，因此解决计算机系统能理解自然语言的问题，引起人们的兴趣和重视，一直是人工智能领域的重要研究课题之一。目前自然语言理解在下列场合获得广泛应用：

1**）**、机器翻译或机助翻译。目前已研制出中、英、日等实用的翻译系统，其翻译的总开销是人工翻译的30％以下。

2**）**、篇章理解。机器阅读，在消化篇章内容的基础上生成其摘要，或回答有关问题。

3**）**、篇章生成。根据以某种形式语言存储在计算机中的信息，生成自然语言。

4**）**、自然语言接口。用户直接采用自然语言，如英语、汉语等同大型数据库或专家咨询系统对话。

**（6）智能信息检索**

数据库系统是存储某个学科大量事实的计算机系统，随着应用的进一步发展，存储信息量愈来愈庞大，研究智能信息检索具有重要的理论意义和实际应用价值。

智能信息检索系统应具有如下的功能：

1）能理解自然语言，允许用户使用自然语言提出检索要求和各种询问；

2）具有推理能力，能根据存储的事实，演绎出所需的答案；

3）系统拥有一定的常识性知识，根据这些常识性知识和专业知识能演绎推理出专业知识中没有包含的答案。

**（7）数据挖掘与知识发现**

数据挖掘(Data Mining)和知识发现(Knowledge Discovery)是20世纪90年代初期新崛起的一个活跃的研究领域。知识发现系统通过各种学习方法，自动处理数据库中大量的原是数据，提炼出具有必然性的有意义的知识，从而揭示出蕴涵在这些数据背后的内在联系和本质规律，实现知识的自动获取。知识发现是从数据库中发现知识的全过程，而数据挖掘则是这个全过程的一个特定的、关键的步骤。数据挖掘的目的是从数据库中找出有意义的模式。这些模式可以是一组规则、聚类、决策树、依赖网络或其他方式表示的知识。一个典型的数据挖掘过程可以分成4个阶段，即数据预处理、建模、模型评估及模型应用。数据预处理阶段主要包括数据的理解、属性选择、连续属性离散化、数据中噪声及丢失值处理、实例选择等。建模包括学习算法的选择、算法参数的确定等。模型评估是进行模型训练和测试，对得到的模型进行评价。在得到满意的模型后，就可以运用此模型对新数据进行解释。

**（8）专家系统**

1976年斯坦福研究所开始开发探矿专家系统PROSPECTOR，1980年首次实地分析华盛顿某山区地质资料，发现了一个钼矿。1981年斯坦福大学研制成功专家系统AM，能模拟人类进行概括、抽象和归纳推理，发现某些数论的概念和定理

**（9）自动程序设计**

自动程序设计的任务是设计一个程序系统，它接受关于所设计的程序要求实现某个目标的非常高级的描述作为其输入，然后自动生成一个能完成这个目标的具体程序。在某种意义上来说，编译程序实际就是去做“自动程序设计”的工作。编译程序接受一段有关干某件事情的源码说明（源程序），然后转换成一个目标码程序（目的程序）去完成这件事情。而这里所说的自动程序设计相当于一种“超级编译程序”，它要求能对高级描述进行处理，通过规划过程，生成得到所需的程序。自动程序设计研究的重大贡献之一是把程序调试的概念作为问题求解的策略来使用。实践已经发现，对程序设计或机器人控制问题，先产生一个代价不太高的有错误的解，然后再进行修改的作法，要比坚持要求第一次得到的解就完全没有缺陷的作法，通常效率要高得多。

**（10）机器人**

第一代：程序控制机器人；

第二代：自适应机器人；

第三代：智能机器人。

2007年9月26日，南京市青少年活动中心科技馆展示大厅里，来自中国科技大学的15个仿人形机器人表演舞蹈《千手观音》、体操表演和赵本山、范伟的小品《卖拐》等三套拿手绝活，达到了我国表演类机器人的最高水平。这批人形机器人具有17个自由度，能够双足行走、前进、后退、转弯、俯卧站立、翻转、鲤鱼打挺、做俯卧撑、站立踢球射门、招手、拥抱，还可以打太极拳、做广播体操、跳舞。用一种锂聚合物电池给它们充电，充一次电可以演出20场。

**（11）组合优化问题**

组合优化问题的求解方法已经应用于生产计划与调度、通信路由调度、交通运输调度、列车组编、空中交通管制和军事指挥自动化等系统。

**（12）人工神经网络**

神经网络已在模式识别、图像处理、组合优化、自动控制、信息处理、机器人学等领域获得日益广泛的应用。

**（13）分布式人工智能**

分布式人工智能(Distributed Artificial Intelligence,简称DAI)是分布式计算与人工智能结合的结果。分布式人工智能系统以鲁棒性作为控制系统质量的标准，并具有互操作性，即不同的异构系统在快速变化的环境中，具有交互信息和协同工作的能力。分布式问题求解把一个具体的求解问题划分为多相互合作和知识共享的模块或者结点，多智能体系统则研究各智能体之间行为的协调。

**（14）智能控制**

智能控制具有两个显著的特点：智能控制是同时具有知识表示的非数学广义世界模型的传统数学模型混合表示的控制过程，也往往是含有复杂性、不完全性、模糊性或不确定性以及不存在已知算法的过程，并以知识进行推理，以启发来引导求解过程。智能控制的核心在高层控制，即组织级控制，其任务在于对实际环境或过程进行组织，即决策与规划，以实现广义问题求解。

**智能控制系统的智能可归纳为以下几个方面：**

先验智能：有关控制对象及干扰的先验知识，可以从一开始就考虑到控制系统的设计中。

反应性智能：在实时监控、辨识及诊断的基础上，对系统及环境变化的正确反应能力。

优化智能：包括对系统性能的先验性优化及反应性优化。

组织与协调智能：表现为对并行耦合任务或子系统之间的有效管理与协调。

**智能控制的开发，目前认为有以下几个途径：**

基于专家系统的专家智能控制。

基于模糊推理和计算的模糊控制。

基于人工神经网络的神经网络控制。

综合以上三种方法的综合型智能控制。

**（15）智能仿真**

智能仿真就是将人工智能技术引入仿真领域，建立智能仿真系统。利用人工智能技术能对整个仿真过程（包括建模、实验运行及结果分析）进行指导，能改善仿真模型的描述能力，在仿真模型中引进知识表示将为研究面向目标的建模语言打下基础，提高仿真工具面向用户、面向问题的能力。

**（16）智能CAD**

智能CAD（ICAD）就是把人工智能技术引入计算机辅助设计领域，建立智能CAD系统。

应用领域：

设计自动化；

智能交互；

智能图形学；

自动数据采集。

从具体技术来看，ICAD技术大致可分为如下几种方法：

规则生成法。

约束满足方法。

搜索法。

知识工程方法。

形象思维方法。

## Python基础

**1 Python简介**

**（1） Python是什么**

相信混迹IT界的很多朋友都知道，Python是近年来最火的一个热点，没有之一。从性质上来讲它和我们熟知的C、java、php等没有什么本质的区别，也是一种开发语言，而且已经进阶到主流的二十多种开发语言的top 5（数据源自最新的TIOBE排行榜）。来头不小啊！二十多种主流的开发语言，我该从哪一个开始呢？人生苦短，let‘s python！

**（2）Python的由来和发展趋势**

Python的前世源自鼻祖“龟叔”。1989年，吉多·范罗苏姆（Guido van Rossum）在阿姆斯特丹为了打发无聊的圣诞节，决心开发一个新的脚本解释程序，自此Python和创始人“龟叔”开始进入公众视野。他希望这个新的叫做Python的语言，能符合他的理想：创造一种C和shell之间，功能全面，易学易用，可拓展的语言。

1991年，第一个Python编译器诞生。它基于C语言实现，并能够调用C语言的库文件。后面历经版本的不断换代革新，Python走到了非常具有里程碑意义的一个节点，即2004的2.4版本诞生了目前最流行的WEB框架Django！六年后Python发展到2.7版本，这是目前为止2.x版本中最新且较为广泛使用版本。

2.7版本的诞生不同于以往2.x版本的垂直换代逻辑，它是2.x版本和3.x版本之间过渡的一个桥梁，以便最大程度上继承3.x版本的新特性，同时尽量保持对2.x的兼容性。因此3.x版本在2.7版本之前就已经问世，从2008年的3.0版本开始，python3.x系呈迅猛发展之势，版本更新活跃，一直发展到现在最新的3.5.2版本。3.5版本也是目前3.x系列中主流且广泛使用的版本，后续相关程序的demo，默认均基于3.5版本展开。

**（3）Python 2 VS Python 3**

许多初学者会纠结于从2.x版本还是3.x版本开始python学习之路。这里简要对比下个人学习了解到的两者的差异：

3.x版本中 PRINT IS A FUNCTION，使用起来更简单方便；

3.x版本中默认使用unicode编码，妈妈再也不用担心你的程序编码问题了；

3.x版本中某些库进行了改名，具体还有待在后续学习中深入展开；

目前3.x版本还不能完全支持Twisted module，这恐怕是3.x版本中最大的遗憾。也不要过于纠结这点，据说官方正在弥补这个不足，同时也相信能有办法解决的：）

In summary : Python 2.x is legacy, Python 3.x is the present and future of the language. 长江后浪推前浪，python的未来是属于3.x系列的，官方宣布2020年之后不再支持2.x系列中最nb的2.7版本，2.x版本将逐渐寿终正寝，最终被铭刻在开发语言的丰碑上。

所以学习3.x版本没错的！

**（4）Python的优缺点**

**@@Python优点**：

1）.“优雅”、“明确”、“简单”

这是python的定位，使得python程序看上去简单易懂，初学者容易入门，学习成本更低。但随着学习的不但深入，python一样可以满足胜任复杂场景的开发需求。引用一个说法，Python的哲学是就是简单优雅，尽量写容易看明白的代码，尽量写少的代码。

2）.开发效率高

python作为一种高级语言，具有丰富的第三方库，官方库中也有相应的功能模块支持，覆盖了网络、文件、GUI、数据库、文本等大量内容。因此开发者无需事必躬亲，遇到主流的功能需求时可以直接调用，在基础库的基础上施展拳脚，可以节省你很多功力和时间成本，大大降低开发周期。

3）.无需关注底层细节

Python作为一种高级开发语言，在编程时无需关注底层细节（如内存管理等）。

4）.功能强大

Python是一种前端后端通吃的综合性语言，功能强大，php能胜任的角色它都能做，至于后端如何胜任，需要在后续学习中逐步领悟。

5）.可移植性

Python可以在多种主流的平台上运行，开发程序时只要绕开对系统平台的依赖性，则可以在无需修改的前提下运行在多种系统平台上，具体有待后续学习中深入展开。其他优点有待继续发掘。

**@@@Python缺点：**

1）代码运行速度慢

因为Python是一种高级开发语言，不像c语言一样可以深入底层硬件最大程度上挖掘榨取硬件的性能，因此它的运行速度要远远慢于c语言。另外一个原因是，Python是解释型语言，你的代码在执行时会一行一行地翻译成CPU能理解的机器码，这个翻译过程非常耗时，所以很慢。而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。

但需要注意的是，这种慢对于不需要追求硬件高性能的应用场合来讲根本不是问题，因为它们比较的数量级根本不是用户能直观感受到的！想想程序执行所需的时间数量级？例如开发一个下载MP3的网络应用程序，C程序的运行时间需要0.001秒，而Python程序的运行时间需要0.1秒，慢了100倍，但由于网络更慢，需要等待1秒，用户体验几乎没有差别，除非你用非常精确的计时器来计时。

2）发布程序时必须公开源代码

什么？有没有搞错？是的，发布程序时我们必须公开源代码！还是因为Python是一种解释性语言，没有编译打包的过程（据说最新的python可以打包，但本质上还是把源代码和解释器打在一起，没有太大实际意义）。想想我们的shell脚本是不是也是这个情况，你能在不发布源代码的情况下发布一个黑盒子程序来让别人正常使用么？

这个缺点仅限于你想单纯靠卖开发出来的软件产品挣钱的时候。但在这个开发互联的时代，不靠卖产品本身来赚钱的商业模式越来越主流了，所以问题也不是没法解决。

总结起来，Python的优点多于缺点，而且缺点在多数它能胜任的情况下不是根本性问题，瑕不掩瑜，尽情享受它给我们带来的乐趣和便利吧！

**（5）Python的适用领域**

1）. 云计算开发

云计算领域最火的语言，典型代表openstack。

2）. web开发

众多优秀的web框架，众多优秀的大型web站点（Youtube,instagrm,豆瓣等）均基于python开发。

3）. 系统运维

各种自动化工具的开发，CMDB、监控告警系统、堡垒机、配置管理&批量分发工具等均可以搞定。这是自己需要重点学习的领域。

4）. 科学计算、人工智能

据网上了解用于围棋大战的谷歌阿发狗就使用了python开发（但估计只是部分而已）。

5）. 图形GUI处理

暂时没多少体会，后续领悟吧。

6）. 网络爬虫

不知道给网络爬虫单独归类是否合适，姑且这样吧。这年头网络爬虫无处不在，很多都是基于python开发，包括谷歌的爬虫。估计这是爬虫工程师的必备工具吧。

**（6）Python应用场景**

谷歌：Google app engine、google earth、爬虫、广告等

Youtube: 世界最大的在线视频网站基于python开发，好强大

Instagram：美国最大的图片分享网站，全部基于python开发

facebook：大量的基础库基于python开发

Redhat：yum包管理工具基于python开发

国内知名互联网公司的应用：豆瓣、知乎的网站，还有很多主流的互联网公司（阿里、腾讯、百度、金山、搜狐、盛大、网易、新浪、网易、果壳、土豆等）通过python来完成各种任务。

**2 Python基础**

**（1）编程开发规范**

1）. 代码开头

python中代码必须顶格编写,除非被作为语句嵌套在条件判断或条件循环中。

2）. 代码层级

代码与代码之间的逻辑层级关系（涉及逻辑判断、包含关系等情况）通过空格来界定，第一层顶格编写，第二层（被包含或者逻辑判断后的执行动作）开头预留4个空格（如果不考虑跨平台可以使用tab）

3）. 对齐方式

同一个层级的代码必须严格对齐，如第一层都是顶格，第二层都是4个空格开头，第三层则是8个空格，以此类推。

4）. 注释

单行注释用#开头，单独一行作为注释或者在代码后面通过#跟上注释均可，多行注释在首尾处用成对的三引号引用即可，可以是成对的三个单引号或者三个双引号。

5）. 代码块

目前来看if逻辑判断和for循环条件后面均以冒号结尾，然后通过4个空格开始下一层级的实际执行动作语句编写。这部分在冒号后面接上的语句称为代码块。

6）. 程序命名

编写的python程序必须以.py结尾。

7）. 项目规范

启动一个项目开发时先在开发工具中创建project，然后根据需要创建目录，最后在目录下创建当前活动的py程序。

**（2）变量**

python中的变量也用于存储临时值以便后续灵活调用，在定义时只能使用字母、数字、下划线的任意组合，且不能以数字开头，不能以python内置的关键字来命名变量（如and，import，global，finally等，有待后续学习加深印象）。对变量进行字符串类型的赋值时，字符串需要用成对的引号引起来，可以是单引号、双引号或者三引号。

建议通过以下两种形式来定义较为复杂的变量，便于阅读：

1）. 驼峰体

即每个单词的首字母大写，其他小写，如MappingTable

2）. 下划线

不同单词之间通过下划线分开，如mappting\_table

**（3）字符编码**

字符编码经历了ASCII码->unicode->UTF-8的演变过程，python 2.x默认使用ASCII编码，而3.x默认使用UTF-8编码。由于字符编码事关重大，一言不合就会导致乱码，因此强烈建议统一使用万用的utf-8编码。怎么使用呢？

1）. 程序开头明确显式定义

#!/usr/bin/env python

# -\*- coding: utf-8 -\*-

windows下编程可忽略第一行，但为了保留移植性建议始终声明变量。

2）. 在程序解释器中选择utf-8字符编码

以上两个条件缺一不可，虽然python 3.x版本默认使用utf-8编码，但还是建议在程序中声明编码格式。

以上是结果，下面谈谈为什么强烈建议使用urf-8编码，以及为什么utf-8编码就可以通吃。

众所周知计算机是美国人发明的，因此最早就只有包括所有字母的大小写、数字、必要的特殊符号等合计才127个字符被编码到计算机里，这个编码就是ASCII编码。由于需要进行编码的字符才有127个，因此用7个二进制位就足以搞定了，但8个二进制位才能构成一个完整的字节，因此索性用8个二进制位来进行编码，剩余的一位只好浪费掉。所以ASCCI码只占用一个字符。

随着计算机技术的推广和普及，其他不以英语为官方语言的国家也开始使用计算机了。那么现在问题来了，一个字节的8个bit位只能表示非常有限的字符（设计时只包括所有字母的大小写、数字、必要的特殊符号，其中整数还不能超过255），既有的ASCII编码表里没有包含其他国家的语言字符，但显然也没法强迫全球都使用英语，于是各个国家开始进行本土化改良，设计能满足自己国家需要的编码，如国内的GB2312。这里的改良涉及两个要点，一是原有的一个字节的编码空间要扩充为两个字节甚至更多，二是新的编码还不能与既有的ASCII编码冲突，否则老美要跟你捉急了！你可以想象，中国搞定了一套，日本也搞定了一套，接着是韩国、越南等巴拉巴拉，大家都有自己的编码标准，各自为政，遇到多语言混合使用的场景这编码表还不炸开锅，连同计算机一起傻掉！这时Unicode应运而生。Unicode把所有语言都统一到一套编码里，用两个甚至更多的字节来进行编码（通常用两个字节，搞不定时再喊人加砝码呀），这样就不会再有乱码问题了。

Unicode搞定了多国语言乱码和编码统一的问题，但你会发现如果需要处理的内容全部是英文的话，用Unicode编码比ASCII编码需要多一倍的存储空间，在存储和传输上就不太经济划算。OK，Unicode的演进改良版utf-8就是来解决这个问题的。utf-8的优势在于它使用“可变长编码”，在继续沿用ASCII编码基础上来对1个字节不能搞定的那些语言字符进行编码设计，这样英文还是用1个字节，遇到中文就用3个或者更多的字节来处理。这样既可以保障空间和性能的最大化，还能兼顾老美的老巢和其他非英文的编码兼容问题，一举三得！所以你会发现ASCII其实是utf-8的子集，utf-8向下兼容。

**（4）Python解释器**

python是一门解释性语言，自然不能缺了解释器（类似于JVM）。官方支持Cpython、Jpython、Ironpython和pypy等，默认使用基于C语言实现的Cpython。它也是最为广泛使用的解释器。程序运行时会把.py文件转换成.pyc字节码文件，然后运行在python虚拟机上。

**（5）构建Python的开发环境**

构建好包括解释器（虚拟机）和开发工具IDE在内的Python的开发环境才便于展开程序开发之旅。其中解释器（虚拟机）从python官方下载安装即可，需要注意的是建议安装3.5版本，windows下安装后需要把python的bin目录和scripts目录追加到系统环境变量path和用户变量path中。我的win 7用户不是以administor，补充追加了用户变量后才能在cmd下傻瓜式地输入python。linux下系统默认是2.x的版本，安装3.5版本时需要先将旧版本重命名为其他目录（如python2.6）后再安装，安装完毕无需额外设置环境变量。好的程序猿都得有自己好使的IDE，python程序建议使用pycharm工具来进行开发。从官网下载安装社区版即可，需要注意的是选择后本地的解释器和项目的默认字符编码。其他一些好玩的特性、快捷键等只有后续逐步体会了。

**（6）模块初识**

python内置了很多模块，同时也可以自定义编写一些好玩的模块，使用模块时需要通过关键字import导入。导入后python会从系统变量的路径中从前往后一次查找，首次成功查找匹配后不再进行后续查找。因此如果需要使用自定义模块，需要特别关注自定义模块的存放位置，底线是只存放在系统变量的一个路径下，否则可能得到意外的结果。

系统变量的路径可通过以下方式查看：

以上是windows下3.5版本的输出结果，linux平台或者其他版本的输出有所差别，以实际为准。

**（7）用户输入和格式化输出**

用户输入通过input来实现，示例如下：

YourName=input("Please input you name here: ")

简单的输出通过print来实现，示例如下：

print("Hi,", YourName) #调用输出上面用户输入的变量YourName

注意：

1）. 3.x版本中input默认接收用户输入的类型为字符串，如果需要使用其他的数据类型则需要进行转换，如 UserAge=int(input("Please input your name:"))

2）. print输出中可以接上转义符号\来实现更多的输出显示效果，如 \n表示换行，\t为制表符。

print的使用进阶——字符串的格式化输出：

python中，对字符串进行格式化输出与c语言是相同的，即借助%来实现。%s表示与字符串替换，%d表示用整数替换，%f表示用浮点数替换，如果输出的字符串中本身就需要%（比如输出比率），则需要用连续的两个%%来输出%。

示例程序如下：

#!/usr/bin/env python

#-\*- coding:utf-8 -\*-

print ("Your name is %s" %"Bob")

print ("Your age is %d" %21)

print ("Your name is %s,age is %d" %("Bob",21))

从以上示例程序可看出，利用print进行格式化输出的基本格式是在需要替换输出的位置用%s,%d等占位符，然后在后面通过%来接上实际替换后需要输出的内容，如果需要替换输出的元素不止一个，则需要在%后面通过一对圆括号把实际替换后需要输出的内容包含进来，同时在每个元素之间用逗号隔开。

**（8）Python的运行逻辑**

Python的运行工作逻辑如下：

内存加载代码->语法分析->解释器加载->生成字节码->执行字节码->生成机器码->CPU执行

**3 开始你的Python开发之旅**

**（1）第一个Python程序--HelloWorld**

python的第一个程序也从hello world开始把：

#!/usr/bin/env python

#! -\*- coding:utf-8 -\*-

print("Hello world!")

执行结果：

"C:\Program Files\Python35\python.exe" D:/python/Day1/test/HelloWorld.py

Hello world!

Process finished with exit code 0

**（2）If-else判断**

1）If-else语法规范

以下是if-else判断的语法结构规范

if condition1:

command\_layer1\_1

if condition2:

command\_layer2\_2

else:

command\_layer2\_2

else:

command\_layer1\_2

以下为一个演示两层if-else循环的程序：

#!/usr/bin/env python #顶格编写

#! -\*- coding:utf-8 -\*-

user\_input = input("Please input you username:")

if user\_input == "Bob": #注意这里的冒号结尾

passwd\_input = input("Please input your password:") #注意从这里开始，第一个if条件为真时需要执行的动作语句均需要左缩进4个空格

if passwd\_input == "password": #第一个if下的第二个if，仍然要左缩进4个空格，同时冒号结尾

print("Welcome login,%s!" %user\_input) #第二层if条件为真时执行的动作语句，需要在第一层语句基础上再缩进4个空格，因此需要缩进8个空格

else: #第二层if-else中的else，因此需要与第二层if对齐，缩进4个空格

print("Invalid username or password, please check your input") #第二层if-else条件为假时执行的动作语句，同样需要与第二层if一样缩进8个空格

else: #第一层if-else中的else关键字，顶格冒号结尾

print("Invalid username or password, please check your input") #第一层if-else判断条件为假时执行的动作，与第一层if一样需要缩进4个空格

说明：该示例程序仅为演示多层if-else的语法结构，程序本身的设计存在漏洞；空格缩进在pycharm IDE环境中会被自动处理，但在普通文件编辑器中需要手动设置。

以下为改良版示例程序，通过引入对if的多条件判断来避免上述程序的漏洞：

1 #!/usr/bin/env python

2 #! -\*- coding:utf-8 -\*-

3 username=input("Please input you username:\n")

4 passwd=input("Please input you password:\n")

5 if username == "Bob" and passwd == "password":

6 print("Welcome login, %s!" %username)

7 else:

8 print("Invalid username or password, please check your input!")

此时只有用户名和密码同时输入正确了才会给出相应提示，否则均提示口令无效，避免暴力破解。

2）if-elif-else扩展

上述判断均为单一式的if-else判断，以下为if-elif-else的判断扩展：

语法结构：

if condition1:

command1

elif condition2:

command2

elif condition3:

command3

else condition4:

command4

不过这种结构仅仅适用于单一条件存在多种case情况下，语法结构看起来还是比较简单，当然顶格、左缩进4个空格和冒号这些规范一样要遵循。

还是来一个示例程序加深理解：

#!/usr/bin/env python

#! -\*- coding:utf-8 -\*-

age=int(input("Please input your age\n"))

if age >= 18:

print("Oh, you're an adult\n")

elif age >=6:

print("Ha, you're a teenager\n")

else:

print("Come on, little kid!\n")

（3）For循环

1）for循环语法规范

For循环的基本语法规范是：

for variable in XXX:

loop command

其中variable表示命名的变量，一般程序中使用i，j等等，XXX表示变化的范围，可以是list列表，一般会考虑使用range函数，来表示一个整数序列，如range（5）就表示小于5的整数序列，即0-4。 语法规范中同样需要for语句后面的结尾冒号，以及循环体中的4个空格的左缩进。

2）示例程序

猜数字游戏，通过系统生成一个随机数作为预设年龄，对用户提供3次猜的机会，前两次如果没有猜中给出数字范围大小的提示，如果第3次还没有猜中则给予鼓励提示，同时打印出这个预设的数字，当然三次当中有任何一次猜中会给用户猜中提示的：

1 #!/usr/bin/env python

2 #! -\*- coding:utf-8 -\*-

3 import random #导入随机数模块

4 Age=random.randrange(10)#随机生成一个小于10的整数（0-9，不包括负数），并赋值给Age

6 for i in range(3):

7 if i < 2:

8 guess\_number=int(input("Please input the age of my dog you guess:\n"))

9 if guess\_number > Age:

10 print("The age you guess is a little big, think smaller!\n")

11 elif guess\_number < Age:

12 print("The age you guess is a little small, think bigger!\n")

13 else:

14 print("Bingo, you got the number,congratulations!\n")

15 break

16 else:

17 guess\_number=int(input("Please input the age of my dog you guess:\n"))

18 if guess\_number == Age:

19 print("Bingo, you got the number,congratulations!\n")

20 else:

21 print("Oh,you just got bad luck, come to try again, you can do it! The actual age of my dog is %d...\n"% Age)

注意：

1.为便于程序调试可先对Age赋值固定的数字

2.前两次如果猜中了就需要退出循环，这里使用了break关键字，意思是跳出整个循环，与之相对应的还有continue关键字，用于跳出当次循环，二者的使用区别还有待进一步深入学习

OK，现在来改进下程序， 当用户连续三次猜错后继续给机会，让用户选择是继续猜还是直接退出，如果继续则再一次获得三次猜的机会，如此循环下去。还是用for循环吧：

1 #!/usr/bin/env python

2 #! -\*- coding:utf-8 -\*-

3 Age=22

4 counter=0

5 for i in range(10):

6 if counter < 3:

7 guess\_number=int(input("Plese input your guess number:\n"))

8 if guess\_number == Age:

9 print("You got the number, congratulations!")

10 break

11 elif guess\_number > Age:

12 print("The number you guessed is too big, guess a smaller one\n")

13 else:

14 print("The number you guessed is too small, guess a bigger one\n")

15 counter += 1

16 elif counter == 3:

17 continue\_flag=input("Do you want to continue? Please type Y to continue or N to quit:\n ")

18 if continue\_flag == "Y":

19 counter = 0

20 else:

21 print("Bye")

22 break

23 else:

24 print("You've tried too many times.")

这里为了保证每一个轮回中的第四次（上述程序中的第四次和第八次）能让程序继续循环，引入了另外一个变量来进行计数并重置。把for循环换作while循环看起来差不多：

1 #!/usr/bin/env python

2 #! -\*- coding:utf-8 -\*-

3 Age=22

4 i=0

5 counter=0

6 while counter < 10:

7 if i < 3:

8 guess\_number=int(input("Plese input your guess number:\n"))

9 if guess\_number == Age:

10 print("You got the number, congratulations!")

11 break

12 elif guess\_number > Age:

13 print("The number you guessed is too big, guess a smaller one\n")

14 else:

15 print("The number you guessed is too small, guess a bigger one\n")

16 i += 1

17 else:

18 continue\_flag=input("Do you want to continue? Please type Y to continue or N to quit:\n ")

19 if continue\_flag == "Y":

20 i = 0

21 else:

22 print("Bye")

23 break

24 counter +=1

**4 搭建开发环境**

1) 可以到www.python.org下载安装包，然后通过configure、make、make install进行安装。

2) 也可以到www.activestate.com去下载ActivePython组件包。(ActivePython是对Python核心和常用模块的二进制包装，它是ActiveState公司发布的Python开发环境。ActivePython使得Python的安装更加容易，并且可以应用在各种操作系统上。ActivePython包含了一些常用的Python扩展，以及Windows环境的编程接口)。对ActivePython来说，如果你是windows用户，下载msi包安装即可；如果你是Unix用户，下载tar.gz包直接解压即可。

3) Python的IDE，包括PythonWin、Eclipse+PyDev插件、Komodo、EditPlus

**5 python小结**

1）. Python是一门主流且发展迅猛的开发语言，目前存在2.x和3.x两个版本分支，前端后端都能胜任，功能强大，在互利网公司应用广泛，值得托付；

2）. Python程序的开发需要安装解释器和IDE，建议使用pycharm，2.x和3.x下的默认字符编码不同，永远推荐使用utf-8编码，包括程序编码和解释的编码两个层面；

3）. 系统自带模块和自定义编写的模块均可以通过import导入使用；

4）. 在进行python程序开发时，注意代码需要分层体现，第一层代码顶格编写，第二层（用在第一次出现的流程控制语句中的代码块）左缩进4个空格；

5）. input用于接收用户输入，默认输入的数据类型是字符串，可通过int关键字转换为整数。print可进行简单输出，如要格式化输出则需要借助%s %d %f等来实现，具体视数据类型而定；

6）. 流程控制语句if-else判断和for、while循环都需要注意代码书写格式，流程控制语句关键字一定是以冒号结尾，具体的流程控制代码块则需要左缩进处理，第一层左缩进4个空格，第二层8个空格，以此类推；

7）.continue和break可分别用于退出当前（当次）流程控制和整个循环，课堂上演示了在for循环中的使用，在if-else判断中不能使用。

## 三、阿里云认证

**课程一、云计算 - 网站建设：部署与发布**

阿里云网站建设认证课程教你如何掌握将一个本地已经设计好的静态网站发布到Internet公共互联网，绑定域名，完成工信部的ICP备案。

**课程二、云计算 - 网站建设：简单动态网站搭建**

阿里云简单动态网站搭建课程教你掌握如何快速搭建一个WordPress动态网站，并会对网站进行个性化定制，以满足不同的场景需求。

**课程三、云计算 - 云服务器管理维护**

阿里云服务器运维管理课程教你掌握快速开通一台云服务器，并通过管理控制台方便地进行服务器的管理、服务器配置的变更和升级、数据的备份，并保证其可以正常运转并按业务需求随时进行配置的变更。

**课程四、云计算 - 云数据库管理与数据迁移**

阿里云云数据库管理与数据迁移认证课程掌握云数据库的概念，如何在云端创建数据库、将自建数据库迁移至云数据库MySQL版、数据导入导出，以及云数据库运维的常用操作。

**课程五、云计算 - 云存储：对象存储管理与安全**

阿里云云储存认证课程教你掌握安全、高可靠的云存储的使用，以及在云端存储下载文件，处理图片，以及如何保护数据的安全。

**课程六、云计算 - 超大流量网站的负载均衡**

掌握如何为网站实现负载均衡，以轻松应对超大流量和高负载。

**课程七、大数据 - MOOC网站日志分析**

本课程可以帮助学员掌握如何收集用户访问日志，如何对访问日志进行分析，如何利用大数据计算服务对数据进行处理，如何以图表化的形式展示分析后的数据。

**课程八、大数据 - 搭建企业级数据分析平台**

模拟电商场景，搭建企业级的数据分析平台，用来分析商品数据、销售数据以及用户行为等。

**课程九、大数据 - 基于LBS的热点店铺搜索**

本课程可以帮助学员掌握如何在分布式计算框架下开发一个类似于手机地图查找周边热点（POI）的功能，掌握GeoHash编码原理，以及在地理位置中的应用，并能将其应用在其他基于LBS的定位场景中。

课程中完整的演示了整个开发步骤，学员在学完此课程之后，掌握其原理，可以在各种分布式计算框架下完成此功能的开发，比如MapReduce、Spark。

**课程十、大数据 - 基于机器学习PAI实现精细化营销**

本课程通过一个简单案例了解、掌握企业营销中常见的、也是必需的精准营销数据处理过程，了解机器学习PAI的具体应用，指导学员掌握大数据时代营销的利器---通过机器学习实现营销。

**课程十一、大数据 - 基于机器学习的客户流失预警分析**

本课程讲解了客户流失的分析方法、流程，同时详细介绍了机器学习中常用的分类算法、集成学习模型等通用技能，并使用阿里云机器学习PAI实现流失预警分析。可以帮助企业快速、准确识别流失客户，辅助制定策略进行客户关怀，达到挽留客户的目的。

**课程十二、大数据 - 使用DataV制作实时销售数据可视化大屏**

帮助非专业工程师通过图形化的界面轻松搭建专业水准的实时可视化数据大屏，以满足业务展示、业务监控、风险预警等多种业务的展示需求。

**课程十三、大数据 - 使用MaxCompute进行数据质量核查**

通过本案例，学员可了解影响数据质量的因素，出现数据质量问题的类型，掌握通过MaxCompute（DateIDE）设计数据质量监控的方法，最终独立解决常见的数据质量监控需求。

**课程十四、大数据 - 使用Quick BI制作图形化报表**

阿里云Quick BI制作图形化报表认证课程教你掌握将电商运营过程中的数据进行图表化展现，掌握通过Quick BI将数据制作成各种图形化报表的方法，同时还将掌握搭建企业级报表门户的方法。

**课程十五、大数据 - 使用时间序列分解模型预测商品销量**

使用时间序列分解模型预测商品销量教你掌握商品销量预测方法、时间序列分解以及熟悉相关产品的操作演示和项目介绍。

**课程十六、云安全 - 云平台使用安全**

阿里云云平台使用安全认证课程教你了解由传统IT到云计算架构的变迁过程、当前信息安全的现状和形势，以及在云计算时代不同系统架构中应该从哪些方面利用云平台的优势使用安全风险快速降低90%。

**课程十七、云安全 - 云上服务器安全**

阿里云云上服务器安全认证课程教你了解在互联网上提供计算功能的服务器主要面临哪些安全风险，并针对这些风险提供了切实可行的、免费的防护方案。

**课程十八、云安全 - 云上网络安全**

了解网络安全的原理和解决办法，以及应对DDoS攻击的方法和防护措施，确保云上网络的安全。

**课程十九、云安全 - 云上数据安全**

了解云上数据的安全隐患，掌握数据备份、数据加密、数据传输安全的解决方法。

**课程二十、云安全 - 云上应用安全**

了解常见的应用安全风险，SQL注入原理及防护，网站防篡改的解决方案等，确保云上应用的安全。

**课程二十一、云安全 - 云上安全管理**

了解云上的安全监控方法，学会使用监控大屏来监控安全风险，并能够自定义报警规则，确保随时掌握云上应用的安全情况。