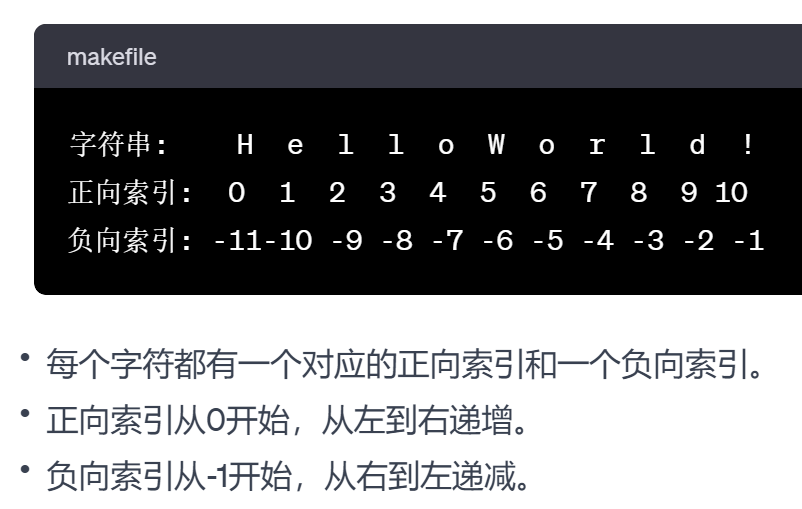
1. 选择题 3’\* 10
2. 定义str = “HelloWorld!”，以下可以输出“World”子串的选项是（ ）
3. print(str[-6 : -1])
4. print(str[-5 : -1])
5. print(str[-5 : 0])
6. print(str[-6])

在字符串 "HelloWorld!" 中，正向和负向索引如下：



对于给出的选项：

A. **print(str[-6 : -1])**：从索引-6 (W) 到 -1 (!) 之前结束，输出 "**World**"。

B. **print(str[-5 : -1])**：从索引-5 (o) 到 -1 (!) 之前结束，输出 "**orld**"。

C. **print(str[-5 : 0])**：这个切片**不会工作**，因为在Python的切片机制中，结束索引0在负向索引-5之前，所以不会输出任何内容。

D. **print(str[-6])**：仅输出索引-6处的字符，即 "**W**"。

因此，只有选项A print(str[-6 : -1]) 可以输出 "World"。

1. Python语言执行min(‘orange’)的结果是（ ）
2. ‘o’
3. ‘e’
4. ‘a’
5. ‘n’

当在Python中使用 min() 函数对字符串进行操作时，这个函数会返回字符串中“最小”的字符。这里的“最小”是根据字符的**ASCII值**（或Unicode值）来确定的。在ASCII编码中，字母按照字典顺序排列，因此字母'a'的ASCII值最小。

在字符串 'orange' 中，字符包括 'o', 'r', 'a', 'n', 'g', 'e'。在这些字符中，根据ASCII值排序，'a' 有最小的ASCII值。因此，min('orange') 返回的是 'a'。这就是为什么正确答案是 C. 'a'。

ASCII（American Standard Code for Information Interchange）是字符编码的标准之一，用于表示文本字符。在ASCII编码中，每个字符被赋予了一个唯一的数字。这些数字代表了不同的字符，例如字母、数字和其他符号。

以下是一些英文字母及其对应的ASCII值的示例：

**'a' 的 ASCII 值是 97**

'b' 的 ASCII 值是 98

'c' 的 ASCII 值是 99

...

'z' 的 ASCII 值是 122

**'A' 的 ASCII 值是 65**

'B' 的 ASCII 值是 66

'C' 的 ASCII 值是 67

...

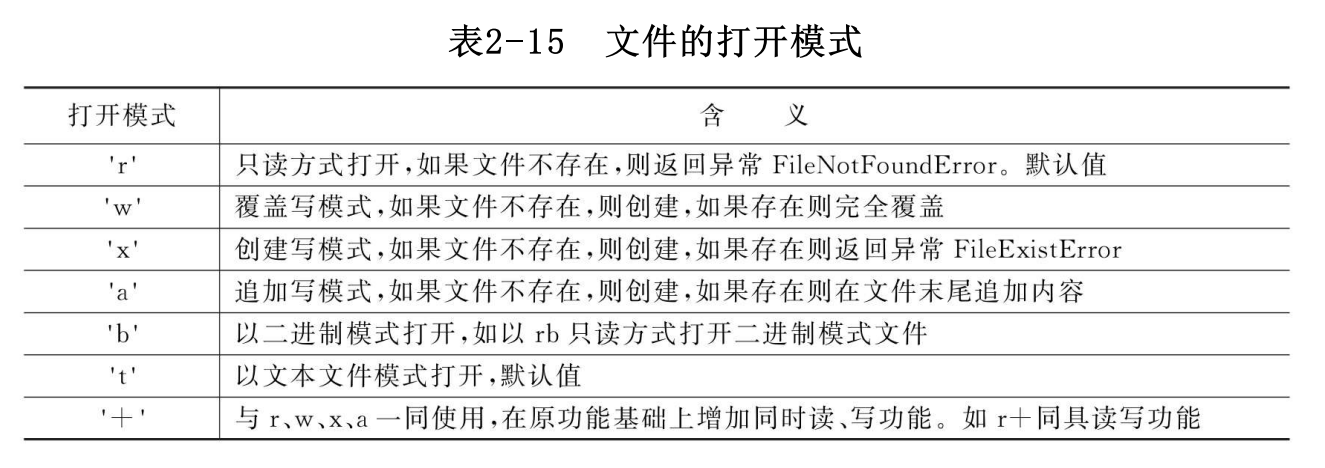
'Z' 的 ASCII 值是 90

小写字母的ASCII值总是大于大写字母。

1. 以下选项中，哪一项不是Python对文件的打开模式（ ）
2. ‘r’
3. ‘+’
4. ‘w’
5. ‘c’

在Python中，文件打开模式指定了文件被打开和操作的方式。这些模式决定了你可以对文件执行哪些操作，比如读取、写入或追加。下面是一些标准的文件打开模式及其含义：

书2.6.1 文件的基本操作



**'r'（只读模式）**： 打开文件进行读取。如果文件不存在，会产生错误。

**'+'（更新模式）**： 与其他模式（如'r+'或'w+'）结合使用时，表示打开文件进行读取和写入。

**'w'（写入模式）**： 打开文件进行写入。如果文件已存在，其内容会被覆盖；如果文件不存在，会创建新文件。

相比之下，'c' 不是Python中的一个标准文件打开模式。因此，在提供的选项中，D选项 **'c' 是不正确的**，不属于Python对文件的标准打开模式。

除了前面提到的几种模式外，Python还提供了其他一些文件打开模式：

**'a'（追加模式）**： 打开文件用于追加。如果文件存在，文件指针会放在文件结尾，新的内容将被写到已有内容之后。如果文件不存在，创建新文件进行写入。

**'b'（二进制模式）**： 与其他模式结合使用（如'rb'、'wb'、'ab+'）表示以二进制格式打开文件。用于非文本文件，如图像或声音文件。

**'x'（排他创建模式）**： 如果文件不存在，创建新文件并打开写入。如果文件已存在，触发FileExistsError。

**'t'（文本模式）**： 与其他模式结合使用（如'rt'或'wt+'）表示文本模式，默认模式。用于处理文本文件。

在实际使用中，可以根据文件类型和需求选择合适的模式组合。例如，'rb' 用于读取二进制文件，'w+' 用于读写文本文件，如果文件不存在则创建。



1. 关于Python组合数据类型，以下选项中描述错误的是（ ）
2. Python的str，tuple和list类型都属于序列类型
3. Python组合数据类型能够将多个同类型或不同类型的数据组织起来，通过单一的表示使数据操作更有序，更容易
4. 组合数据类型可以分为3类：序列类型，集合类型 和 映射类型
5. 序列类型是二维元素向量，元素之间存在先后关系，通过序号访问

在提供的选项中，描述错误的是 D。

A 正确。Python 的字符串（str）、元组（tuple）和列表（list）都属于序列类型。

B 正确。Python 组合数据类型确实可以组织多个同类型或不同类型的数据。

C 正确。Python 组合数据类型可以分为序列类型、集合类型和映射类型。

D 错误。序列类型不一定是二维元素向量。它们是一维结构，其中的元素具有确定的顺序和索引，但并不限于二维。因为：

**序列类型并非仅限于二维元素向量**：Python中的序列类型，如字符串（str）、列表（list）和元组（tuple），是可以包含任意数量维度的元素的一维结构。它们主要特点是元素有序排列，并可以通过索引进行访问。这些序列可以包含各种类型的数据，包括但不限于数字、字符串或其他序列（比如列表中的列表，实现多维数据结构）。

**序列类型的一维性质**：尽管可以在序列类型中嵌套其他序列来创建类似于二维或多维数组的结构，但序列类型本身是一维的。例如，一个列表可以包含多个子列表，每个子列表代表二维数组中的一行，但这并不改变列表本身是一维序列的事实。

因此，将序列类型描述为“二维元素向量”是不准确的。它们更准确地描述为可以索引、有序排列的一维元素集合。



1. 下列哪个表达式在Python中是非法的（ ）
2. x = y = z = 1
3. x = (y = z + 1)
4. x, y = y, x
5. x += y

在Python中，非法的表达式是 B. x = (y = z + 1)。原因如下：

A. x = y = z = 1：这是**链式赋值**。在Python中，你可以同时为多个变量赋予相同的值。这行代码将1赋值给z，y和x。

B. x = (y = z + 1)：这是非法的。在Python中，赋值操作（如y = z + 1）不能作为另一个赋值操作的一部分。Python中赋值语句不返回任何值，因此不能像表达式那样用在其他赋值语句中。

C. x, y = y, x：这是**变量交换**的有效写法。Python允许使用**元组解包**（tuple unpacking）来交换变量的值，无需额外的临时变量。

D. x += y：这是**增量赋值**操作符的使用，等价于x = x + y。它在原有的x值上加上y的值，然后将结果赋值给x。



1. 关于Python的lambda函数，以下选项中错误的是（ ）
2. lambda函数将函数名作为函数的结果返回
3. f = lambda x, y : x + y执行后，f的数据类型为数字类型
4. lambda用于定义简单的，能够在一行内表示的函数
5. 可以使用lambda函数定义列表的排序原则

在提供的选项中，错误的是 B。

A. 正确。Lambda函数是一个匿名函数，它返回函数对象本身，而不是具有显式名称的函数。

B. 错误。f = lambda x, y: x + y 执行后，f 的数据类型是函数，而不是数字类型。f 可以被调用来执行加法操作。

C. 正确。Lambda函数通常用于创建简单的一行函数。

D. 正确。Lambda函数经常用于定义排序操作的关键字，例如在列表的 sort() 方法或 sorted() 函数中。



1. 长短期记忆网络（LSTM）的提出者是（ ）
2. Jürgen Schmidhuber
3. Ian Goodfellow
4. Geoffrey Hinton
5. Michael L. Jordan

长短期记忆网络（LSTM）的提出者是 A. Jürgen Schmidhuber，以及他的学生 Sepp Hochreiter。他们在1997年共同发表了关于LSTM的论文，提出了这一重要的循环神经网络架构。因此，正确答案是 A.

A. **Jürgen Schmidhuber**: 他是LSTM（长短期记忆网络）的共同发明者之一。Schmidhuber的研究领域广泛，包括神经网络、深度学习、数字物理学等。除了LSTM之外，他还在人工智能领域有其他多项贡献。

B. **Ian Goodfellow**: 最著名的是他在深度学习领域的工作，尤其是**对生成对抗网络**（GANs）的发明。Goodfellow的这一贡献极大地推动了机器学习领域的发展，尤其是在图像生成和处理方面。

C. **Geoffrey Hinton**: 被广泛认为是深度学习的先驱之一。他在神经网络领域的研究极为深入，包括**反向传播算法**和**深度信念网络**等。Hinton的工作对人工智能的复兴起到了关键作用。

D. **Michael I. Jordan**: 他在机器学习领域有着深远的影响，特别是在统计学习理论方面。Jordan的研究涵盖了广泛的主题，包括概率图模型、贝叶斯方法等。他是当代机器学习和统计数据科学领域的领军人物之一。

1. 以下哪个领域不属于新一代人工智能的驱动因素（ ）
2. 大数据
3. 新算法
4. 计算力
5. 无线网

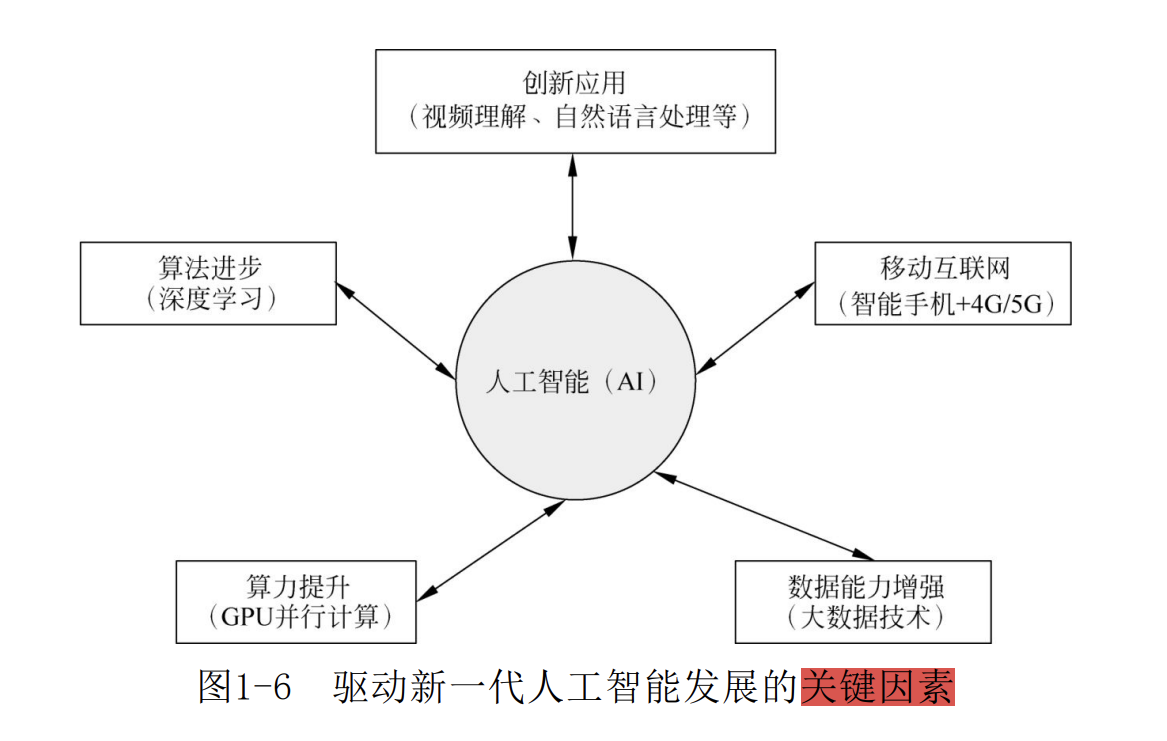
在提供的选项中，不属于新一代人工智能的驱动因素的是 D. 无线网。

A. **大数据**：大量的数据是训练复杂机器学习模型的基础，对人工智能的发展至关重要。

B. **新算法**：算法创新，特别是在机器学习和深度学习领域，是推动人工智能进步的关键因素之一。

C. **计算力**：更强的计算能力，包括GPU和其他专用硬件的发展，使得复杂的人工智能算法成为可能。

D. **无线网**：虽然无线网络技术是现代通信的重要部分，但它并不直接驱动人工智能技术的发展，尤其是与大数据、算法创新和计算力相比。书1.5 新一代人工智能的驱动因素



1. 以下哪个不属于典型的深度学习模型（ ）
2. 卷积神经网络
3. 随机森林
4. 循环神经网络
5. 深度置信网络

在提供的选项中，不属于典型的深度学习模型的是 B. 随机森林。

A. **卷积神经网络（CNN）**：是深度学习中用于图像处理、视频分析等领域的一种主要模型。特别适用于处理具有网格状拓扑结构的数据，如图像（2D网格）和视频（3D网格）。CNN通过卷积层来提取空间特征，广泛应用于图像识别、视频处理等领域。

B. **随机森林**：是一种机器学习模型，属于集成学习方法，但不是深度学习模型。它通过组合多个决策树来提高预测准确性和控制过拟合，常用于分类和回归任务。

C. **循环神经网络（RNN）**：是处理序列数据（如时间序列或自然语言）的深度学习模型。RNN能够处理不同长度的输入序列，通过隐藏层的状态保持前一个时间步的信息。

D. **深度置信网络（DBN）**：是一种由多层神经网络构成的深度学习模型，常用于特征学习和分类任务。DBN是一种由多层限制玻尔兹曼机（RBM）或自编码器构成的深度神经网络。DBN在无监督学习中被用来逐层预训练网络，以帮助改善后续监督学习任务的性能。

1. 在 1- 2000 的整数中随机地取一个数，取到的整数既不能被6整除，又不能被8整除的概率是（ ）
2. 0.25
3. 0.5
4. 0.75
5. 0.8

为了计算在1到2000的整数中随机取一个数，这个数既不能被6整除也不能被8整除的概率，我们可以采取以下步骤：

**1、计算能被6或8整除的数的数量**：首先，找出在1到2000之间能被6整除的数的数量，然后找出能被8整除的数的数量。由于6和8不是互质的，我们还需要减去既能被6也能被8整除的数的数量，以避免重复计算。

**2、计算既不能被6也不能被8整除的数的数量**：从总数2000中减去能被6或8整除的数的数量。

**3、计算概率**：将既不能被6也不能被8整除的数的数量除以总数2000，得到概率。让我们进行计算。

找出能被6整除的数的数量：在1到2000的整数中，能被6整除的数的数量是 total\_numbers // 6。

找出能被8整除的数的数量：在1到2000的整数中，能被8整除的数的数量是 total\_numbers // 8。

找出同时能被6和8整除的数的数量：既能被6也能被8整除的数实际上是能被24（6和8的最小公倍数）整除的数，数量是 total\_numbers // 24。

计算既能被6或8整除的数的总数量：这是能被6整除的数和能被8整除的数的总和，减去同时能被6和8整除的数的数量，即 divisible\_by\_6 + divisible\_by\_8 - divisible\_by\_24。

计算既不能被6也不能被8整除的数的数量：这是总数2000减去能被6或8整除的数的数量。

计算概率：这是既不能被6也不能被8整除的数的数量除以总数2000。通过这个过程，我们得到既不能被6整除也不能被8整除的数的概率是0.75。

如果需要**手动计算**这个概率，可以按照以下步骤操作：

计算能被6整除的数字数量：在1到2000之间，每6个数就有一个能被6整除。所以数量是 **2000÷6=333**（不完全整除时，取整数部分）。

计算能被8整除的数字数量：类似地，数量是 **2000÷8=250**。

计算同时能被6和8整除的数字数量：这些数字其实是能被24（6和8的最小公倍数）整除的。数量是 **2000÷24=83**。

计算既能被6或8整除的数字数量：这是前两步的总和减去第三步的结果，即 **333+250−83=500**。

计算既不能被6也不能被8整除的数字数量：从总数2000中减去上一步的结果，即 **2000−500=1500**。

计算概率：这是既不能被6也不能被8整除的数字数量除以总数2000，即 **1500÷2000=0.75** 或 **75%**。

1. 判断题 3` \* 10
2. 数据量的爆发性增长是驱动新一代人工智能的唯一因素（ ）❌

错误。数据量的爆炸性增长是推动新一代人工智能发展的重要因素之一，但并非唯一因素。其他因素还包括算法创新、计算能力的提升等。

1. 虚拟现实和增强现实都是新型视听技术且两者不是等价的（ ）✔️

正确。**虚拟现实（VR）**和**增强现实（AR）**都是新型视听技术，但两者不是等价的。VR提供完全虚拟化的环境，而AR则是在真实世界中增加虚拟元素。

1. 知识图谱与自然语言处理是两个相同的人工智能的关键技术（ ）❌

错误。知识图谱和自然语言处理是人工智能领域中的两个不同的关键技术。知识图谱用于组织和处理结构化知识，而自然语言处理专注于理解和生成自然语言。

1. 深度学习中的Wake-Sleep算法是由Hinton提出的（ ）有争议❌✔️

深度学习中的Wake-Sleep算法实际上是由Geoffrey Hinton和他的学生Peter Dayan以及Brendan Frey共同提出的。该算法主要用于训练深度信念网络（Deep Belief Networks, DBNs），并且是处理无监督学习问题的一种方法。因此，虽然Hinton确实参与了该算法的开发，但它并非他独立提出的。

1. Python代码的注释只有一个种方式，即注释行前加 “#” 符号（ ）❌

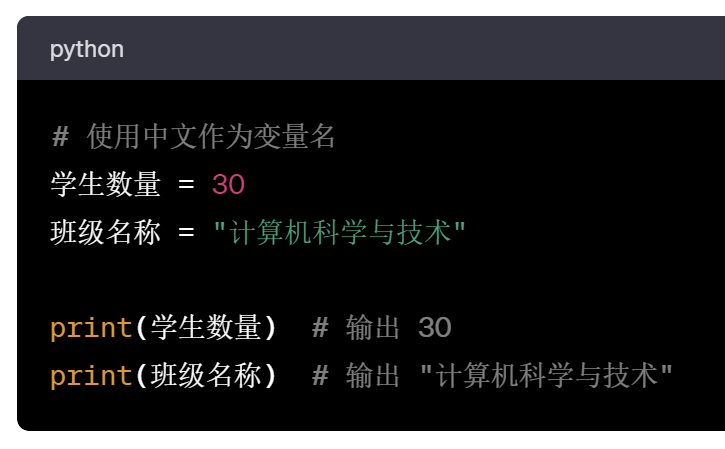
错误。在Python中，注释不仅可以通过在行前加上 “#” 符号实现**单行注释**，还可以使用三重引号（''' 或 """）来创建**多行注释**。以下是两种注释方式的例子：



1. Python 3.0中，可以使用中文作为变量名（ ）❌

正确。在Python 3.0及更高版本中，你可以使用中文作为变量名。这是因为Python 3支持Unicode，允许几乎所有Unicode字符作为变量名的一部分。以下是使用中文作为变量名的示例：

在这个例子中，`学生数量` 和 `班级名称` 是用中文命名的变量。这样的命名在Python 3中是完全有效的。



1. Python表达式 ‘a’ + 1的值是 ‘b’（ ）❌

错误。在Python中，尝试将字符串 'a' 与整数 1 相加会导致类型错误，因为Python不支持将字符串和整数直接相加。例如：

图形用户界面, 应用程序

描述已自动生成

如果你想要通过字符的ASCII值将 'a' 转换为 'b'，你需要先将 'a' 转换为其ASCII值，然后加1，最后将结果转换回字符：

图形用户界面

低可信度描述已自动生成

在这个例子中，ord('a') 返回字符 'a' 的ASCII值（97），加1后得到98，然后 chr(98) 将数字转换回对应的字符 'b'。

1. 5人排成一列，甲乙二人不相邻的概率为3/5 （ ）✔️

正确。要计算5人排成一列时甲乙二人不相邻的概率：

**1、计算总的排列数**：5个人排成一列的总排列数是5的阶乘，即 **5!=5×4×3×2×1=120**。

**2、计算甲乙相邻的排列数**：将甲乙看作一个单元，那么有4个单元（甲乙单元和其他3个人）。这4个单元的排列方式有 **4!=4×3×2×1=24** 种。甲乙内部可以互换位置，因此每种排列方式有2种变化（甲在前或乙在前）。因此，甲乙相邻排列总数是 **24×2=48**种。

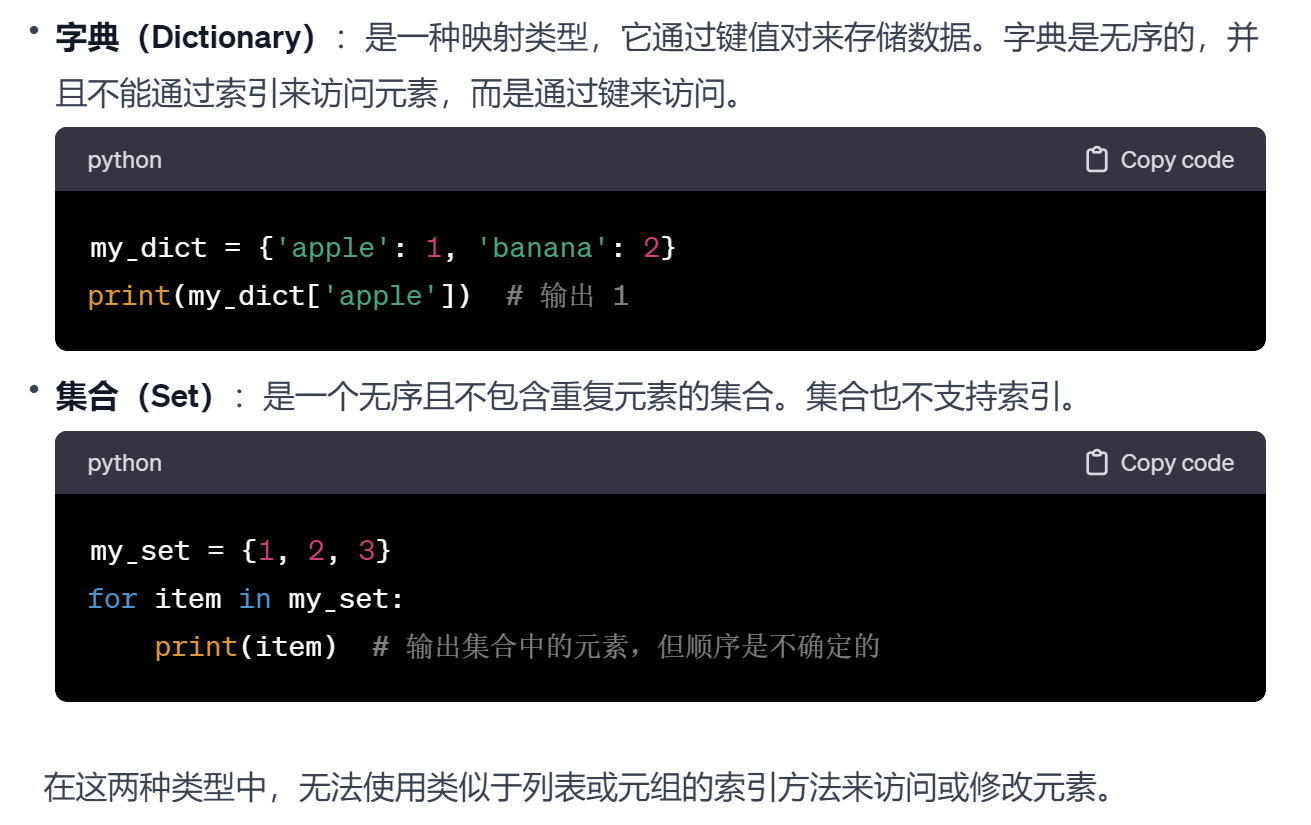
**3、计算甲乙不相邻的排列数**：从总排列数中减去甲乙相邻的排列数，即**120−48=72**。

**4、计算概率**：甲乙不相邻的概率是不相邻的排列数除以总排列数，即 **72÷120=0.6 或 60%**，也就是**3/5**

因此，5人排成一列，甲乙二人不相邻的概率是**3/5**。

1. 字典和集合都属于无序序列，都支持双向索引（ ）❌

错误。在Python中，字典和集合都是无序的集合类型，但它们不支持索引，更不支持双向索引。



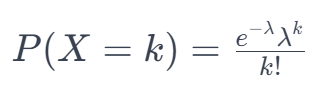
1. 对于服从泊松分布的随机变量，其数学期望与方差相等，而且知道数学期望或者方差后即能完全确定其分布（ ）✔️

正确。泊松分布是一种描述在固定时间或空间内发生某事件次数的概率分布。对于服从泊松分布的随机变量，其特征包括：

**数学期望（均值）和方差相等**：泊松分布的参数 λ 表示单位时间或单位空间内事件平均发生的次数，同时也是该分布的均值和方差。即对于泊松分布，如果 X∼Poisson(λ)，则 E(X)=Var(X)=λ。

通过期望或方差确定分布：由于泊松分布的均值和方差相等，因此知道其中一个就可以确定分布的参数 λ，从而完全确定该泊松分布。

数学上，泊松分布的概率质量函数（PMF）表示为：



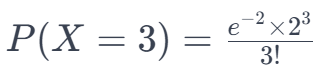
其中 ，k 是非负整数，表示事件发生的次数，λ 是分布的参数，表示事件的平均发生率。

关于泊松分布的实际例子：

假设某个快递站平均每小时收到2个包裹。我们想要知道在一个小时内收到恰好3个包裹的概率。

在这个例子中，收到的包裹数量可以看作是一个服从泊松分布的随机变量 X。给定的平均数（λ）是2，所以泊松分布的参数也是2。

我们要计算的概率可以表示为 P(X=3)，根据泊松分布的公式，它可以计算为：



这就是在一个小时内收到恰好3个包裹的概率。

1. 填空题 2`\* 10
2. 假设从一大批产品的100个样品中，可获得一级品60个，则这批产品的一级品的置信度为0.95的置信区间为（ ）

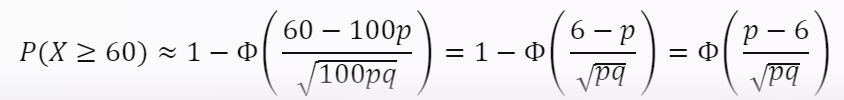
这个问题涉及的是二项分布参数的区间估计，因为我们在处理的是一系列的是/非（即一级品或非一级品）结果。具体步骤如下：

1、设总体的一级品率为 p，则每个样品的一级品的概率为 p，不是一级品的概率为 **q = 1−p**。

2、设 X 为100个样品中一级品的个数，由于每个样品相互独立，X 服从二项分布 B(100,p)，其数学期望为 **E(X) = 100p**，方差为 **D(X) = 100pq**。

3、由题意，要求一级品率的置信度为 0.95，即 P(X ≥ 60) ≥ 0.95，其中 60 是样本中一级品的个数。

4、利用棣莫弗-拉普拉斯中心极限定理，当 N 很大时，可以近似地认为 X 服从正态分布 N(100p,100pq)，于是有

其中 Φ(x) 是标准正态分布的分布函数，满足 Φ(−x) = 1−Φ(x)。

5、由于 Φ(x) 是单调递增的，要使上式不小于 0.95，只需使 钟表的特写

中度可信度描述已自动生成，其中 Φ−1(x) 是 Φ(x) 的反函数。

6、查标准正态分布的数值表，可得 Φ−1(0.95) ≈ 1.645，故有 图片包含 图标

描述已自动生成，解得 p≥0.5。

7、因此，一级品率的置信度为 0.95 的置信区间为 [0.5, 1]。

1. 一个复杂系统由N个相互独立起作用的部件所组成，且至少有80%的部件工作才能使整个系统正常工作，那么如果每个部件的可靠性（即部件正常工作的概率）为0.9，则N至少为（ ）时，才能使得系统的可靠性不低于0.95

这个问题可以用棣莫弗-拉普拉斯中心极限定理来解决。具体步骤如下：

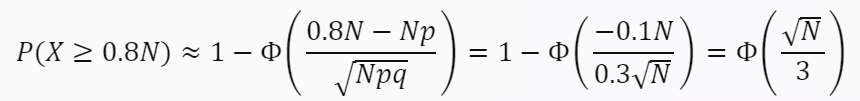
1、设系统中有 N 个部件，每个部件正常工作的概率为 p=0.9，则每个部件不工作的概率为 ****。

2、设 X 为系统中正常工作的部件的个数，由于每个部件相互独立，X 服从二项分布 图片包含 文本

描述已自动生成，其数学期望为 ****，方差为 ****。

3、由题意，要求系统的可靠性不低于 0.95，即 ****，其中 0.8N 是系统正常工作所需的最少部件数。

4、利用棣莫弗-拉普拉斯中心极限定理，当 N 很大时，可以近似地认为 X 服从正态分布 ****，于是有

其中 Φ(x) 是标准正态分布的分布函数，满足 卡通人物

低可信度描述已自动生成。

5、由于 Φ(x) 是单调递增的，要使上式不小于 0.95，只需使 钟表的特写

中度可信度描述已自动生成，其中  是的反函数。

6、查标准正态分布的数值表，可得 ，故有 ，解得 。

7、因此，N 至少为 25 时，才能使得系统的可靠性不低于 0.95。

1. 样本中所含个体的个数，称为（ ）；在进行抽样的，样本的选取必须是随机的

样本中所含个体的个数称为样本容量。在进行抽样时，确保样本的选取是随机的非常重要，这样可以保证样本能够更好地代表总体，使得从样本中得出的结论更加可靠和具有代表性。随机抽样方法有助于减少偏差，并使得样本统计量的推断更加准确。

1. 已知字符串str = “2024!”，则print(str[: : -1])结果为（ ）

执行 print(str[::-1]) 结果是将字符串 str = "2024!" 进行反转。在Python中，切片操作 str[start:end:step] 中，当 step 为负数时，表示**反向索引**，它指定了从字符串的尾部向头部取值的方向，step 的绝对值表示每次跳过的字符数。

因此，如果 step 是 -1，那么每个字符都被取到，结果是字符串的完全反转，str[::-1] 会从字符串的末尾开始反向遍历整个字符串。所以，对于 str = "2024!"，执行 print(str[::-1]) 的结果将是 "!4202"。

如果 step 是 -2 或其他更大的负数，将会跳过一些字符。例如，step 为 -2 时，将取字符串中每两个字符的**第一个**。



1. 执行print(2 \*\* 4 + 16 % 3)结果为（ ）

执行 print(2 \*\* 4 + 16 % 3) 的结果可以通过计算得出：

2 \*\* 4 是2的4次方，即 2×2×2×2=16。

16 % 3 是16除以3的余数，即 16 mod 3=1。

所以结果是 16+1=17。

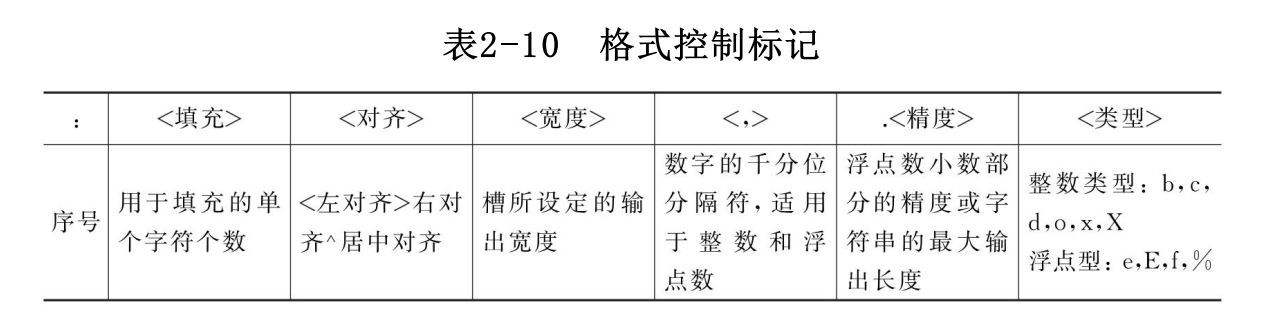
因此，执行 print(2 \*\* 4 + 16 % 3) 的结果为 17。

1. 执行print(“{:& > 8}”.format(‘python’))结果为（ ）

执行 print("{:&>8}".format('python')) 的结果将是字符串 'python' **右对齐**，并且在左侧用 & 字符填充至总长度为8个字符。具体输出将是 "&&python"。

这里，{:&>8} 指定了格式化选项，其中 & 是填充字符，> 表示右对齐，而 8 指定了字符串的总长度。由于 'python' 已有6个字符，因此在左侧将添加2个 & 符号以达到总长度8个字符。

书表 2 -10 格式控制标记



1. 执行print(‘:’.join(‘1,2,3,4,5’.split(‘,’)))结果为（ ）

执行 print(':'.join('1,2,3,4,5'.split(','))) 的结果将是：

首先，'1,2,3,4,5'.split(',') 会根据逗号 , 分割字符串，得到一个包含字符串 '1', '2', '3', '4', '5' 的列表。接着，':'.join(...) 会使用冒号 : 将这个列表中的元素连接起来。因此，最终的输出将是 "1:2:3:4:5"。



split() 函数将字符串分割成一个列表，而 join() 函数则将列表中的元素合并成一个字符串。这是因为 split() 根据指定的分隔符（在这个例子中是逗号 ,）来切割原始字符串，并返回一个由切割得到的子字符串组成的列表。随后，join() 函数取这个列表中的每个元素，并在它们之间插入指定的连接符（这里是冒号 :），最终生成一个新的字符串。这就是列表如何被转换成字符串的过程。

1. 基于大数据知识工程的知识图谱是（ ）学派人工智能的代表性应用成果

基于大数据知识工程的知识图谱是符号主义（Symbolic AI）学派人工智能的代表性应用成果。符号主义学派强调使用明确的符号和逻辑规则来表示知识和推理过程。知识图谱作为一种组织和利用知识的方法，依赖于明确的符号化表示和关系定义，非常符合符号主义的方法论。

1. 计算机视觉可分为计算机成像学，（ ），三维视觉，动态视觉和视频编解码五大类

计算机视觉可以分为计算成像学、图像理解、三维视觉、动态视觉和视频编解码这五大类。图像理解涉及到对图像内容的分析和解释，例如物体识别、场景重建和图像分割等，这是计算机视觉领域的一个重要分支。

1. 机器学习中常用的学习规则有Hebb学习规则，Delta算法，（ ）算法

机器学习中常用的学习规则除了Hebb学习规则和Delta算法外，还包括反向传播（Backpropagation）算法。

反向传播算法：是神经网络中最为关键的学习规则之一，特别是在深度学习和多层神经网络的训练中发挥着重要作用。该算法通过计算损失函数关于网络参数的梯度，并以此来更新网络中的权重和偏差，从而优化模型性能。

Hebb学习规则：这是神经网络学习中的一种基本规则，常被总结为“神经元同时激活则它们之间的连接加强”。简而言之，如果两个神经元的活动是相关的，那么连接它们的突触权重就会增加。

Delta算法：也称为最小均方算法（LMS），是一种简单的基于梯度下降的学习规则。它用于调整神经网络权重，以最小化网络输出和期望输出之间的差异。Delta算法特别适用于单层前馈神经网络。

1. 简答题 10’\* 4
2. 欠拟合和过拟合是所有机器学习算法都要考虑的问题，请写出5种常见的过拟合解决办法

**权值衰减**：在神经网络训练过程中，通过乘以一个小于1的因子（例如0.9，通常称为衰减率）来减小权重的绝对值。这种方法通过惩罚过大的权重值，有助于减少模型的复杂度和过拟合风险。

**合适的停止训练标准**：通过早停法（Early Stopping）来避免过拟合。这涉及在训练过程中监控模型在验证集上的表现，当在验证集上的性能不再提高时停止训练。

**保留验证数据集**：将数据分为训练集和验证集，使用验证集来监测模型的泛化能力。这有助于检测和防止过拟合。

**获取额外数据进行交叉验证**：使用交叉验证方法，如k-折交叉验证，来评估模型的泛化能力。这涉及将数据分成多个部分，并轮流使用其中一部分作为测试集，其他部分作为训练集。

**正则化**：在目标函数或代价函数（损失函数）中加入正则项（如L1或L2正则化），以惩罚大的权重值。L1正则化倾向于生成稀疏权重矩阵，而L2正则化倾向于更平滑的权重调整。正则化项的系数（通常称为 lambda 或 alpha）控制了正则化的强度。这有助于控制模型复杂度，防止过拟合。

1. 卷积神经网络结构主要包括哪些层？请说明Softmax函数的功能，若设C表示神经网络输出层的输出数量，i表示输出层的第i个输出，Qi表示第i个输出值，使用上述符号，写出Softmax函数的数学表达式

卷积神经网络（CNN）结构主要包括以下几种层：

**卷积层（Convolutional Layer）**：使用一组可学习的过滤器（卷积核）来提取输入数据的空间特征。

**激活层（Activation Layer）**：通常使用非线性激活函数，如ReLU，为网络引入非线性特性。

**池化层（Pooling Layer）**：通过对特征图进行下采样，降低特征维度，减少计算量和参数防止过拟合。

**全连接层（Fully Connected Layer）**：CNN末端，将前一层输出展平后连接到每个神经元，用于学习非空间特征，将特征映射到输出。

**归一化层（Normalization Layer）**：如批归一化，用于加速训练。这类层（如批归一化）主要用于调整前一层激活的均值和方差，以帮助稳定和加速神经网络的训练过程，减少所谓的内部协变量偏移。它们并不专注于输出层，而是可以应用于网络中的多个层。归一化层关注于训练过程中的稳定性和效率。

**Softmax层**：通常作为网络的最后一层。Softmax函数用在神经网络输出层的函数，尤其是在多分类问题中。函数将输出转换为概率分布，确保所有输出值的总和为1。Softmax关注于将最终输出转换为可解释的概率。

**Softmax激活函数**的功能是将一个向量转换成概率分布。在CNN中，它常用于输出层，将神经网络的输出转化为概率分布。其数学表达式为：文本

描述已自动生成

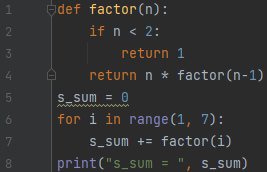
在这个表达式中，图片包含 图标

描述已自动生成是Softmax函数的输出，表示第 i个输出的概率。图片包含 文本

描述已自动生成是神经网络输出层的第 i 个输出值，而 C 代表输出层的总输出数量。分母中的求和项 卡通人物

中度可信度描述已自动生成是对所有输出值的指数进行求和，确保了Softmax输出的总和为1，形成一个有效的概率分布。

1. 分析以下程序，请写出factor(n)的数学表达式，并写出s\_sum的计算表达式，计算过程和print函数的输出结果



factor(n)的数学表达式是n!，表示n的阶乘，即n×(n−1)×(n−2)×…×1。

s\_sum的计算表达式是将1到6（包括6）的每个整数的阶乘求和。因此，s\_sum的计算过程如下：

s\_sum = 1! + 2! + 3! + 4! + 5! + 6! = 1 + 2 + 6 + 24 + 120 + 720 = 873

所以，print函数的输出结果是："s\_sum = 873"。

1. 人工神经元网络中激活函数又叫激励函数，1）简述激活函数的作用；2）写出Sigmoid激活函数，双曲正切激活函数，ReLU激活函数的表达式

1）激活函数的作用：

激活函数在人工神经网络中起着至关重要的作用。它们的主要作用是引入非线性因素到网络中。这种非线性使得神经网络能够处理复杂的任务，如语音识别、图像识别等。如果没有激活函数，无论网络有多少层，输出都只是输入的线性组合，这极大地限制了网络的表达能力和复杂度。激活函数还可以帮助控制神经元的输出范围，并通过梯度下降等方法帮助网络学习和逼近任何复杂的函数关系。

2）常见的激活函数表达式：

·Sigmoid激活函数：图形用户界面

低可信度描述已自动生成

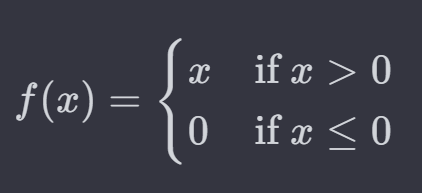
Sigmoid函数将输入值压缩到0和1之间，它非常适合用作输出层在二分类问题中表示概率。

·双曲正切激活函数（tanh）：图形用户界面, 文本, 应用程序

描述已自动生成

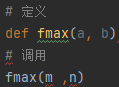
tanh函数也将输入值压缩，但是范围是-1到1，它是Sigmoid函数的缩放版本。

·ReLU激活函数：文本

描述已自动生成或

ReLU（Rectified Linear Unit）函输入值如果是正数，输出本身，否则输出0。ReLU函数的优势在于它的简单性，这导致它在正向传播和反向传播时计算速度很快，而且减少了梯度消失问题的发生。不过，ReLU函数也有所谓的“死亡ReLU”问题，即当神经元的输入始终为负时，神经元可能永远不会被激活，导致信息无法通过网络的某些部分传递。为了解决这个问题，研究者们提出了ReLU的一些变种，如Leaky ReLU或Parametric ReLU（PReLU）。

1. 计算题 10’ \* 3
2. 如图所示，结合函数fmax的定义与调用，1）简述Python中的参数传递机制；2）举例5种参数传递方式



1）Python中的参数传递机制：

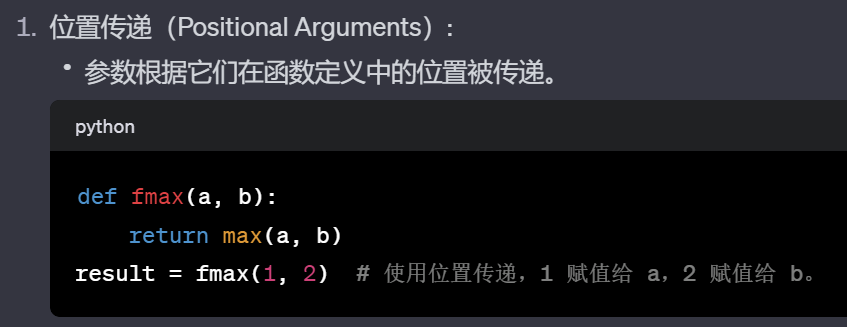
Python中函数的参数传递是递是指将实参传递给形参的过程。函数的定义即函数功能的实现过程，包含函数头与函数体两部分。

函数头括号内是函数的参数列表（formal parameter list），其中的参数是形式参数，简称为“形参”，是调用该函数时传递给它的值，可以是0个，也可以是一个或多个。当传递多个参数时，各参数由逗号分隔。没有参数也需要保留括号。形参只在函数体中有效。

函数被定义好之后不会立即执行，只有被程序调用时才会执行。调用时，参数列表中给出实际要传入函数内部的参数，这类参数称为实际参数，即“实参”。在函数被调用时将实参复制给函数的形参。

2）参数传递的5种方式：

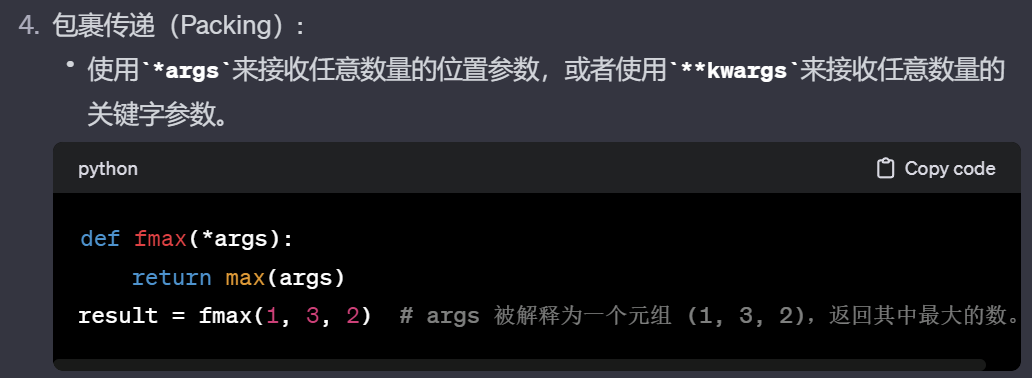
Python中函数支持以多种方式传递参数，包括位置传递、关键字传递、默认值传递、包裹传递、解包裹传递及混合传递。



图形用户界面, 文本, 网站

描述已自动生成 文本

描述已自动生成

 文本

描述已自动生成

文本

描述已自动生成

文本

描述已自动生成

 文本

描述已自动生成 文本

描述已自动生成

1. 将一个骰子掷两次，若两个事件定义为：A=“第一次掷得点数2或5”，B=“两次点数之和至少为7”，求事件A和事件B的概率，并分析事件A和事件B是否相互独立

分析事件A和事件B，并验证它们是否相互独立。

**事件A的概率 P(A):**

事件A是“第一次掷得点数2或5”。在一个公正的六面骰子上，得到任意特定点数的概率是。因此，事件A的概率是得到2或5的概率之和：



**事件B的概率 P(B):**

事件B是“两次点数之和至少为7”。我们可以通过计算所有可能的点数组合，并找出那些点数和至少为7的组合来计算这个概率。总共有 6×6=36 种可能的组合，因为骰子有6个面，每次投掷都有6种可能的结果。那些满足事件B条件的组合有21种，具体为：(1, 6), (2, 5), (2, 6), (3, 4), (3, 5), (3, 6), (4, 3), (4, 4), (4, 5), (4, 6), (5, 2), (5, 3), (5, 4), (5, 5), (5, 6), (6, 1), (6, 2), (6, 3), (6, 4), (6, 5), 和 (6, 6)。因此，事件B的概率是：图形用户界面

中度可信度描述已自动生成

**事件A和B同时发生的概率 P(A∩B):**

事件A和B同时发生意味着第一次掷得2或5，并且两次掷得的点数之和至少为7。满足这两个条件的组合是：(2, 5), (2, 6), (5, 2), (5, 3), (5, 4), (5, 5), 和 (5, 6)。一共有7种组合。因此，事件A和B同时发生的概率是：图形用户界面

中度可信度描述已自动生成

**独立性验证:**

如果事件A和事件B是相互独立的，那么它们同时发生的概率应该等于它们各自概率的乘积：文本

描述已自动生成

根据我们的计算，事件A和B同时发生的概率是 ，而事件A的概率是 ，事件B的概率是 。将事件A和B的概率相乘：，这与事件A和B同时发生的概率相等，所以我们可以确认事件A和事件B是相互独立的。

1. 设随机变量 X1，X2，…，Xn，相互独立且同服从正态分布N(1, 9)，则当n -> +∞时，Yn = 1/n ，依概率收敛于B，试求出B的值

