入学 Python 编程能力测试及预修资料

开课吧人工智能学院

祝贺你， 你已经马上要成为一名真正的算法工程师，不，准确得说，应该是一名人工智能工程师了。而且我们将要进行的是最有趣，最具有挑战性，也是使用最广的自然语言理解方向或者计算机视觉方向。文字和图像是我们日常输入最多的两种信息，希望大家已经做好了准备。

但是，第一步，为了使得我们之后的课程能够合理继续，需要对大家的编程水平进行测试。请大家在以下三道题**任选**一道，完成编码并且测试输入和输出，观察程序的运行是否符合程序的预期。

如果这三道题目对你而言都太过困难，你可以先自学预修课程，我们在测试题的附录为大家提供了预修资源建议。 这些课程都是公开、免费的课程资料，希望大家能够做出来我们的先修测试之后，再继续上课。

可能你已经注意到了，我们的这三道题目，两道是关于数值计算/模拟计算的， 一道是关于语言解析的。 这三道题目每一道都是很有趣的，也是我们以后工作中会经常遇到的问题的一个缩影， 希望大家能花时间解决掉。不论你是选择的自然语言处理还是计算机视觉，这些题目你都可以做。这些程序考察的是编程能力，和具体的背景知识无关。

那现在还有一个问题，如何知道测试题是否做对呢？我们的每道题给大家都给出来足够的测试用例，请确保你在同样的输入下，你的程序和我们给出的测试用例是一样的。第三题是一个随机函数，这个除外。如果你的程序给出的结果和我们给出的参考答案一致，说明你的程序是正确的。如果，你对自己的程序不够有信心不确定是否正确，请及时联系微信小助手，或者在我们系统邮箱里回复进行咨询。我们的邮箱是 ai-college@kaikeba.com

好的， 开始我们的激动的学习之旅吧!

0. Programming Environment:

Python 3.6 (Recommended), Python 2.7

1. Spiral Memory

You come across an experimental new kind of memory stored on an infinite two-dimensional grid.

Each square on the grid is allocated in a spiral pattern starting at a location marked 1 and then counting up while spiraling outward. For example, the first few squares are allocated like this:

```
17  16  15  14  13
18   5   4   3  12
19   6   1   2  11
20   7   8   9  10
21  22  23---> …
```

While this is very space-efficient (no squares are skipped), requested data must be carried back to square 1 (the location of the only access port for this memory system) by programs that can only move up, down, left, or right. They always take the shortest path: the Manhattan Distance between the location of the data and square 1.

For example:

Data from square 1 is carried 0 steps, since it's at the access port.
Data from square 12 is carried 3 steps, such as: down, left, left.
Data from square 23 is carried only 2 steps: up twice.
Data from square 1024 must be carried 31 steps.
How many steps are required to carry the data from the square identified in your puzzle input all the way to the access port?

How to test your answer:

If you input: **100000** Your Answer should be: **173**
If you input: **2345678** Your Answer should be: **1347**

2. Simple Number Finding

You are playing a card game with your friends. This game in China named "扎金花". In this game, the 2, 3, 5 are some simple powerful numbers. Because the combination of 2,3,5 is less than any other combinations but greater than the AAA, which is the king in this game. In today, you want to find if a number is a simple number, in which their factors only include 2, 3 and 5.

So your task is to find out whether a given number is an amazing number.

E.g

Input: 6

Output: (2, 3)

Explanation: 6 = 2 x 3

Input: 8

Output: (2, 2, 2)

Explanation: 8 = 2 x 2 x 2

Input: 14

Output：None
Explanation: 14 is not amazing since it includes another prime factor 7.

How to check your answer:

If you test **1845281250**, your program should give **(2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 5, 5, 5, 5, 5, 5)**
If you test **3690562500**, your program should give **(2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 5, 5, 5, 5, 5, 5);**
If you test **1230187500**, your program should give **(2, 2, 3, 3, 3, 3, 3, 3, 3, 3, 3, 5, 5, 5, 5, 5, 5)**;
If you test **10023750**, your program should give **None**;

3. Random Chinese Sentence Generator

Writing a programming which could generate random Chinese sentences based on one grammar.

Your input grammar is:

```
simple_grammar = """
sentence => noun_phrase verb_phrase
noun_phrase => Article Adj* noun
Adj* => null | Adj Adj*
verb_phrase => verb noun_phrase
Article => 一个 | 这个
noun => 女人 | 篮球 | 桌子 | 小猫
verb => 看着 | 听着 | 看见
Adj => 蓝色的 | 好看的 | 小小的 | 年轻的
"""
```

Your task is define a function called *generate,* if we call generate('sentence'), you could see some sentences like:

>> generate("sentence")

Output: 这个蓝色的女人看着一个小猫

>> generate("sentence")

Output: 这个好看的小猫坐在一个女人

好了，看完这些题目，如果你已经有了想法，知道怎么做，那真是太好了，你已经具备了基础条件，就差一些具体的 AI/机器学习算法知识了。

如果你的程序运行结果与我们的给出的例子一致（第三题是随机函数，应该产生类似的句子），那么请及时联系我们的课程微信助手，她会发送具体的注册、付费链接给你。如果同学要申请奖学金，请在联系小助手的时候，把相关的个人简历或者其他证明材料一起发送给她。

附录-1： Python 自学建议

各位同学大家好，相比各位同学来参加我们的课程都是希望能够获得一些有价值的知识，随着现在 AI 知识的越来越多，我们学习 AI 知识，也不能仅仅浅尝辄止，停留在最基础的编程和工具的使用。但是，基本的编程能力是我们进一步学习的必要保障。但是我们的课程目前不提供基础的编程学习，因为，我们在此为大家如果学习编程尤其是 Python 编程提供一些经验和指导。

大家都知道，程序设计语言这个说法，为什么这个叫做“语言”呢？ 其实所有的编程语言和英语、法语、汉语一样，既然都叫语言，肯定是有一些相似之处的。我们很多同学学习语言的时候都经历过“从入门到放弃”的过程，主要的原因就是学习方法不对。现在大多数地方的学习方式，先给大家很多基础概念、语法规则，然后让大家反复练习那些没有什么实际意义的练习题，就和当年大家学英语希望通过背词典、背语法书来学好英语一样。这样的结果往往是磨灭了学习的热情，而且往往学习效果也不好。

和学外语一样，大家学习一个语言要坚持4个原则：

1. 问题驱动：我们学习的时候，一定是要面向一个有意义或者有趣的问题去解决它，在不断的尝试中，逐渐加强自己的能力；
2. 持续练习：拳不离手曲不离口，大家要学好一个语言，就要 不断的写，多联系；
3. 主动搜索：这个是很重要但是往往被大家忽略的，就是要锻炼自己的搜索能力。我们做算法工作，之后遇到的问题大概率都是新问题，一定要联系自己查找资料，解决问题的能力。 不能养成思维惰性，遇到问题第一反应就是问别人，这样的话，一定是不能成为优秀的算法工程师的。
4. 锻炼思维：我们学编程，重要的是要学习编程的思维体系和方法，所谓的语法特性，其实是最不重要的，现在互联网很发到，一搜就能搜到，我们要练习的是看到一个问题，如果抽象成一个程序能够解决的问题，然后编程实现。

基于以上原则，我给大家推荐一些普遍反应比较良好的课程、资料，推荐资料里边英文课程很多，但是如果不习惯，要多练习练习，以后从事算法工程师的工作，一手的资料都是英文的。

1. https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-0001-introduction-to-computer-science-and-programming-in-python-fall-2016/ 这个课程来源于 MIT(麻省理工)，是 MIT 新生的第一课；
2. https://classroom.udacity.com/courses/cs101 这个是有优达学城的 计算机科学入门它的优点是具有比较强的交互性，而且问题引入模型比较好；
3. https://www.coursera.org/learn/python?action=enroll 这个是密西根大学的 Python 入门课程，也是广受好评；
4. https://www.liaoxuefeng.com/wiki/1016959663602400 这个课程是廖雪峰老师的博客，是我们推荐给大家的唯一一个中文资料，大家遇到不懂的问题，可以查阅；
5. https://composingprograms.com/ 这个是加州大学伯克利大一学生的课程资料，这个内容比较多值得大家花费 2-3 个月仔细学习；
6. http://norvig.com/21-days.html 这个是 Google 研究总监写的关于编程学习的一点经验。