# 卡迪夫计算机科学与信息学院

## 课程作业评估形式

**模块代码：** CMT120型

**模块标题：** 编程基础

**讲师：** 费德里科·利伯拉托雷

**评估标题：** 编程挑战

**定档日期：** 十月23 2023 **提交日期及时间：** 2023年12月14日上午9：30 **回程日期：** 一月15 2024

**如果您已获得延期，则提交截止日期和退货日期将晚于上述日期。当您的延期获得批准时，您将被告知修改后的提交截止日期。**

**如果您推迟秋季或春季学期的评估，您可能会不及格，并且必须重新参加失败或推迟的科目。**

**如果您因情有可原的情况而获得延期，那么您将在下一个预定的评估期内接受评估，在该评估期间，您将接受评估。**

**如果您推迟了秋季或春季评估，并且有资格进行夏季重考，您将在夏季重考期间完成延期评估。**

**如果您被要求留级或在重考期间推迟了复修，您将在下一学年完成评估。**

**作为一般规则，学生在夏季评估期间只能重修 60 个不及格学分（见学术规则第 3.4 节** [**）。**](https://intranet.cardiff.ac.uk/students/study/your-rights-and-responsibilities/academic-regulations) **超过60个学分（本科课程不超过100学分，研究生课程不超过105学分）的学生将被要求留级。此规则有一些例外，它们在考试委员会根据具体情况适用。**

**如果你是硕士生，请注意，推迟评估可能会影响你的论文开始日期。这是**

**因为你必须通过所有教学模块，然后才能开始你的论文。如果您是海外学生，任何延误都可能对您的签证产生影响，特别是如果您打算在完成课程后申请毕业后工作签证。**

**注意：夏季重考期很短，工作人员的支持将很少。因此，如果评估次数很多，这可能是一个紧张的工作时期。**

该作业占该模块总分的 40%。如果课程作业迟交（并且没有情有可原的情况）：

1. 如在截止日期后24小时内提交评估，评估分数将以最低及格分数为上限;
2. 如果在截止日期后 24 小时以上提交评估，则评估将打 0 分。

课程作业提交日期的延期**只能**使用特殊情况程序[提出申请](https://intranet.cardiff.ac.uk/students/study/exams-and-assessment/extenuating-circumstances)。只有经证实有情有可原的学生才能使用情有可原的提交截止日期。在初始提交截止日期之后提交的任何课程作业，如果没有**获得批准**的情有可原的情况，将被视为逾期。

有关情有可原程序的更多信息，请访问内联网：

[https://intranet.cardiff.ac.uk/students/study/exams-and-assessment/](https://intranet.cardiff.ac.uk/students/study/exams-and-assessment/extenuating-circumstances)  [情有可原的情况](https://intranet.cardiff.ac.uk/students/study/exams-and-assessment/extenuating-circumstances)

[https://intranet.cardiff.ac.uk/students/study/your-rights-and-responsibilities/](https://intranet.cardiff.ac.uk/students/study/your-rights-and-responsibilities/academic-regulations)  [学术规定](https://intranet.cardiff.ac.uk/students/study/your-rights-and-responsibilities/academic-regulations)

提交此任务即表示您接受以下声明的条款：

*我在此声明，我提交的作品（或我在小组提交的情况下对它的贡献）都是我自己的作品，它以前没有提交过评估，并且我没有故意允许它被其他学生复制。我声明，除非允许，否则我没有未经授权使用 AI 聊天机器人或工具来完成这项工作。我明白，通过将另一位作家的作品冒充自己的作品来欺骗或试图欺骗考官是剽窃。我也明白，剽窃他人的作品或故意允许其他学生剽窃我的作品是违反大学规定的，这样做将导致分数损失和可能的纪律处分。*

# 分配

要完成本课程，您必须完成一组 Python 和 JavaScript 编程挑战。

每个挑战最多可获得 10 分。因此，完美解决五个练习会给你 50 分（及格），完美解决所有练习会给你 100 分。只有当同时用 Python 和 JavaScript 提交高质量的函数式代码时，才能完美地解决练习。仅使用一种编程语言提供高质量的函数式代码会导致较低的分数（即 10 分中的 5 分）。因此，您仍然可以通过只完成一种语言的问题，或者用两种语言完成一半的问题来通过课程作业。

您可能无法解决所有练习。 *这很好*。我们并不期望你们都能用这两种语言解决所有挑战。但是，您应该能够用一种或两种语言解决足够多的练习，以便能够通过评估并证明您已经达到了所评估的学习成果。

下面将详细描述这些挑战，并且还为您提供了一组测试用例，用于检查您的代码是否生成所需的输出。特别是，每次练习将获得两个测试用例。您应该确保您提交的代码通过提供的测试，以确保其正常运行。但是，请注意，您的代码将针对另外两个不同的测试用例进行测试，而您尚未获得这些用例。然后，每个练习总共将针对四个测试用例进行测试，包括提供的两个测试用例。因此，您应该确保尝试涵盖所有可能的输入，并且您的代码仍然正常运行。您的代码需要通过所有 4 项测试（2 项已看到，2 项未见）才能获得该功能的满分。请注意，每次测试的时间限制为 3 秒。如果您的代码未在时间限制内提供正确答案，则测试失败。

### 完成挑战的说明

* 您可以在 Learning Central 上找到作业的模板代码。 这提供了两个文件夹， python 和 js。在每个文件夹中，您 都会找到一个 模板。js/py} 文件，您应该在其中完成解决方案。您还将发现一个 test\_template。js/py} 文件，其中包含将检查代码功能的测试用例，以及某些测试所需的测试数据文件夹。您也得到供应

带有一个 Readme.md 文件，其中包含有关如何运行测试用例以检查代码的详细说明。

* 在模板中，给出了函数的接口，但函数的主体是空的。通过正确填写函数的主体来解决练习。
* 禁止更改函数的接口。但是，可以定义新功能来支持练习的解决方案。这些函数的名称必须与模板中已有的名称不同。
* 不允许导入任何其他模块。使用模数函数将导致相应练习的零分。
* 在所有练习中，您可以假设输入以适当的格式和类型提供。因此，不需要错误检查。
* 最终提交不应包含任何 输入、 打印或 控制台.log 语句。

在规定时间内，您将因在两种编程语言中解决每个问题而获得分数。解决方案风格和质量将获得进一步的评分。标记方案将在后面进一步详细描述。

## 练习 1：最小分数项

完成函数“exercise1”，该函数将两个正整数作为其参数。参数表示分数的分子和分母。该函数将分数简化为最低项，然后将约化分数的分子和分母作为整数返回。 **Python：** 分子和分母应以元组形式返回。

**JavaScript：** 分子和分母应作为 2 项数组的元素 0 和 1 返回。

### 例子：

* exercise1（12,15） 在 Python 中返回 （4， 5），在 JavaScript 中返回 [4,5]。
* exercise1（8,4） 在 Python 中返回 （2,1），在 JavaScript 中返回 [2,1]。

## 练习 2：神奇的日期

魔术日期是日期乘以月份等于年份最后两位数字的日期。

例如，1960 年 10 月 6 日是一个神奇的日期，因为 6 乘以 10 等于 60，等于 1960 的最后两位数字。

完成函数“exercise2”，该函数分别采用三个正整数（日、月和年）作为其参数，如果日期是魔术日期，则返回 True，否则返回 False。

### 例子：

* exercise2（10,6,1960） 在 Python 中返回 True，在 JavaScript 中返回 true。
* exercise2（10,6,1970） 在 Python 中返回 False，在 JavaScript 中返回 false。

### 练习 3：查找所有子列表

子列表是构成较大列表一部分的列表。子列表可以由单个元素、多个元素组成，也可以完全不包含任何元素。

**示例：[**1]、[2]、[3] 和 [4 ] 都是 [1， 2， 3， 4] 的子列表。这

列表 [2， 3] 也是 [1， 2， 3， 4] 的子列表，但 [2， 4] 不是

[1， 2， 3， 4] 因为元素 2 和 4 在较长的列表中不相邻。空列表是任何列表的子列表。因此，[] 是 [1， 2， 3， 4] 的子列表。列表是其自身的子列表，这意味着 [1， 2， 3， 4] 是

也是 [1， 2， 3， 4] 的子列表。

使用上述子列表的定义，完成函数“exercise3”，该函数将列表作为其唯一参数，并返回一个包含输入列表的每个可能的子列表的列表。

### 例子：

* exercise3（[1,2,3,4]） 返回

[[]，[1]，[2]，[3]，[4]，[1,2]，[2,3]，[3,4]，[1,2,3]，[2,3,4]，[1,2,3,4]]。

* exercise3（['a'，2，（0，“zero”）]） 返回

[[]，['a']，[2]，[（0，“zero”）]，['a'， 2]，[2， （0，“zero”）]，['a'， 2， （0，“zero”）]];

返回的列表元素的顺序并不重要。但是，每个列表中元素的顺序应反映原始列表中的顺序。例如，['a'， 2] 是正确的，而 [2， 'a'] 则不正确。

### 练习 4：英语到猪拉丁语翻译器

猪拉丁语是一种语言游戏或语言游戏，其中英语单词被改变，通常是通过添加一个捏造的后缀，或者通过将一个单词的开始或初始辅音或辅音簇移动到单词的末尾，并添加一个发声音节来创建这样的后缀（维基百科）。

以下规则用于将英语翻译成猪拉丁语：

* 如果单词以辅音（包括“y”）开头，则单词开头的所有字母，直到第一个元音（不包括“y”），都将被删除，然后添加到单词的末尾，然后是“ay”。 **示例：“**  computer”变为“omputercay”，“think”变为“ink-thay”。
* 如果单词以元音开头（不包括“y”），则“way”将添加到单词的末尾。

**示例：“**  algorithm”变为“algorithmway”，“office”变为“officeway”。

完成函数 'exercise4'，该函数将字符串作为唯一的参数，并返回一个表示其 Pig 拉丁语翻译的字符串。

### 例子：

* exercise4（'algorithm'） 返回 'algorithmway'。
* Exercise4（'Computer'） 返回 'omputercay'。

该函数应正确处理大写字母和标点符号，例如逗号、句点、问号和感叹号。您可以假设只有第一个字母可以是大写的，标点符号只能在单词的末尾。

**示例：**如果一个英语单词以大写字母开头，那么它的猪拉丁语表示也应该以大写字母开头，移动到单词末尾的大写字母应该改为小写。例如，“Computer”应变为“Omputercay”。如果单词以标点符号结尾，则在执行转换后，标点符号应保留在单词末尾。例如，“Science！”应该变成“Iencescay！”。

### 练习 5：摩尔斯电码编码器

摩尔斯电码是电信中用于将文本字符编码为两种不同信号持续时间的标准化序列的方法，称为点和

破折号（或 dits 和 dah）。摩尔斯电码以电报发明者塞缪尔·莫尔斯（Samuel Morse）的名字命名（维基百科）。

完成函数“exercise5”，该函数将一串字母和数字作为唯一参数，并返回一个带有摩尔斯电码的字符串。

使用句点“.”表示点，使用减号“-”表示破折号。从字母和数字到破折号和点的映射如下图所示。

一个 U

B V

C W

D X

E Y

F Z

G H I J

K 1

L 2

M 3

N 4

O 5

P 6

Q 7

R 8

S 9

T 0

该函数应在每个破折号和点序列之间留出一个空格，并且应忽略任何非字母或数字的字符。

### 例：

* exercise5（'你好，世界！ 返回 '.... . .-...-.. --- .-- --- .-..-. '.
* 练习4（'Dracarys!!!'） 返回 '-...-..- -.-..- .-.-.-- '.

*提示：* 使用字典来表示摩尔斯电码表。

### 练习 6：拼出数字

完成函数“exercise6”，该函数将 0 到 999 之间的整数作为其唯一参数，并返回包含该数字的英语单词的字符串。

* Exercise6（21） 返回 'twenty-one'。
* Exercise6（191） 返回 'a hundred and 91'。

*提示：*使用一个或多个字典来实现解决方案，而不是使用大型 if/elif/else 构造。

### 练习 7：没有注释的函数

完成函数“exercise7”，该函数将包含文件名的字符串作为其唯一参数。文件名应为源文件。该函数读取文件，识别前面没有注释的函数，并返回其名称列表（表示为字符串）。

**Python：** 为了本练习的目的，假设任何以“def”开头的行，后跟一个空格，都是函数定义的开始。此外，假设当函数具有注释时，注释字符“#”将是上一行的第一个字符。

**JavaScript：** 在本练习中，假设任何以“function”开头的行，后跟一个空格，都是函数定义的开头。此外，假设当函数具有注释时，单行注释符号“//”将是上一行的第一个字符。

### 练习 8：对齐任何文本

完成函数“exercise8”，采用两个参数，文件名和最大长度（即严格的正整数）。该函数打开文件，读取每一行，并返回一个字符串列表，其中每个字符串表示一行，该行在不超过给定最大长度的情况下尽可能多地填充。

**示例：** 考虑一个文件，其中包含“爱丽丝梦游仙境”中的以下行。

爱丽丝是

开始厌倦了坐在姐姐身边

在岸边，无所事事：有一两次，她偷看了她 姐姐 正在读的书，但 它确实 有

不

里面有图片或对话，“一本书有什么用，”爱丽丝想，“没有图片或对话？

length = 50 的相应输出为：

[“爱丽丝开始厌倦了 坐在岸边，被她姐姐坐在岸边，无所事事”，“有一两次她偷看书了”，

“她姐姐在读书， 但里面没有图画，或者，”里面有对话，“”书“有什么用，”爱丽丝想，“没有图片，或者” 对话？

您不需要处理多个文本段落。也就是说，您可以将不同段落中的单词组合在一起。最后，您可以假设没有单词的长度超过最大长度。

### 练习 9：骑士的挑战

完成函数“exercise9”，它有三个参数：初始位置、最终位置和移动次数。如果空棋盘上的骑士最多可以在给定的步数内从起始位置到达最终位置，则该函数返回 True;否则，该函数返回 False。有用的事实：

* 棋盘是一个 8x8 的方形棋盘。
* 棋盘的每个单元格都由其坐标标识：从“a”到“h”的字母标识列，以及标识行的 1 到 8 的数字。使用这种格式向函数提供位置。
* 棋盘只包含骑士，骑士位于指定的初始位置。
* 骑士可以垂直移动两个方格，水平移动一个方格，或者水平移动两个方格，垂直移动一个方格。

### 例子：

* exercise9（'a1'， 'c5'， 2） 返回 True。
* exercise9（'c6'， 'h1'， 1） 返回 False。

*提示：* 您可能希望使用递归。

### 练习 10：物种之战

完成函数“exercise10”，该函数将相同长度的字符串列表作为输入，表示矩形网格。列表中的字符表示网格的单元格。单元格可以采用三个可能的值：

* “X”， 代表物种 X 的个体。
* “O”， 代表物种 O 的个体。
* '.'， 表示空单元格。

### 例：

[“X.......”， "........", ". O"]

是一个 3x8 网格，位置 （1,1） 为“X”，位置 （3,8） 为“O”。

列表中提供的配置表示具有两个竞争物种的环境的当前状态。每个单元格都与其八个相邻单元相互作用，这些相邻单元是水平、垂直或对角线相邻的单元。该函数应根据以下规则返回表示环境下一个状态的字符串列表：

* 如果一个空细胞被至少两个相同物种的个体包围，它就会变成非空的。特别是，它成为附近最常见的物种之一。如果物种之间出现平局，则细胞保持空。
* 如果非空像元被六个以上的非空像元包围，则该像元变为空，无论其种类如何。
* 如果一个非空细胞被少于三个物种的成员包围，则该细胞变为空。
* 如果一个非空单元格被相反物种的成员多于其物种的成员所包围，则该单元格变为空。
* 在任何其他情况下，单元格不会更改其值。

您可以假定字符串输入列表的格式正确。

**示例：** 初始环境

[“XX......”， “XX....O.", ". 噢

成为

[“XXX.....”， “xxx..OOO“， ”XX. O."]

## 评估的学习成果

* LO1：使用高级编程语言完成编程任务。
* LO2：展示对编程概念、简单数据结构和算法的熟悉程度。

## 评估标准

学分将根据以下标准授予。

每项练习最多可获得 10 分。因此，成功解决五个练习会给你 50 分（及格），完美解决十个练习会给你 100 分。

每个练习都有功能（最高 8 分）和风格/质量（最高 2 分）的标记。不是真正尝试解决所提出的问题的练习将获得零分。

**功能 [8 分/练习最多]** 子任务的功能部分由脚本自动标记，这些脚本针对一组测试用例运行已完成的功能。每次测试将为您提供两个测试用例。在评分过程中，每个练习将针对四个测试用例进行测试，包括提供的两个测试用例。测试应在三秒内完成，否则将被视为失败。对于每种语言（Python 和 JavaScript）和练习，通过一项测试将奖励您 1 分。因此，只有当所有测试都以两种语言通过时，才会给出 8 的最大功能标记。

**代码质量和风格 [最多 2 分/练习]**  练习的每个版本（即 Python 和 JavaScript）都根据以下标准进行独立评估：

|  |  |
| --- | --- |
| 高品质和风格（50-100%，  每次练习 0.5-1 分） | 低质量和风格（0-50%，0-0.5  每次练习的分数） |
| 代码是优雅的  代码没有冗余 代码被很好地注释  代码是完全模块化的（即，定义了适当的函数和/或类）  代码巧妙地利用了内置  语言功能和类。 | 代码杂乱或过于冗长  代码具有多个冗余和重复  代码缺少有意义的注释  代码杂乱无章  代码不使用语言  特征 |

因此，仅用一种语言解决的练习最多可以达到一种风格/质量标记。要获得这两个分数，必须用两种语言解决练习。此外，只有通过至少两项测试的练习才会被评估风格和质量。

## 对未来学习的反馈和建议

对您的课程作业的反馈将解决上述标准。反馈和分数将在返回日期通过学习中心和/或电子邮件返回。这项任务的反馈将对你的第二个项目任务有用，也将与任何未来的编程有关。

任务。

## 投稿 须知

所有课程作业都应通过上传到 Learning Central 的方式提交。

|  |  |  |
| --- | --- | --- |
| **描述** | **类型** | **名字** |
| *Python 代码* | 1 .py 文件 | [学号].py |
| *JavaScript 代码* | 1 .js 文件 | [学号].js |

将 [学号] 替换为您的卡迪夫用户名，通常为字母“**c”**（或**“d”**）+ 您的学号，例如 *c1234567*。

提交的任何代码都将在与提供给学生的笔记本电脑相当的系统上运行，并且必须按照上述说明的规定提交。代码应运行，无需对提交的代码进行任何更改，包括编辑文件名。

任何偏离上述提交说明（包括提交文件的数量和类型）都可能导致从总分中扣除最多 10% 的费用。

教职员工保留邀请学生参加会议讨论课程作业提交的权利。

## 支持评估

有关评估的问题可以在第 3-11 周的辅导讲座中 https://stackoverflow.com/c/comsc/ 提出并标记为“[CMT120]”，以及

在临时实际支持会议中。需要注意的是，将仅提供一般指导。解决练习、定义逻辑和调试代码（以及其他）的任务是评估的重点，因此也是学生的责任。

**COMSC：代码中引用指南。**

卡迪夫大学制定了不公平的实践程序，该程序涵盖了如果学生的行为方式“可能为他/她自己或他人获得未经许可的优势或高于他/她的能力所能获得的分数或成绩”，会发生什么。本文档中的指南旨在帮助学生避免这种情况，应与以下人员一起阅读：

* 学生内联网上提供的有关不公平做法的信息：https://intranet.cardiff.ac.uk/students/study/exams-and-assessment/sitting-your 考试/作弊和不公平做法
* 卡迪夫大学关于不公平做法的规定： [https://intranet.cardiff.ac.uk/intranet/students/documents/exams-and-assessment/1.11-](https://intranet.cardiff.ac.uk/intranet/students/documents/exams-and-assessment/1.11-Unfair-Practice-Procedure.pdf) [不公平-做法-程序.pdf](https://intranet.cardiff.ac.uk/intranet/students/documents/exams-and-assessment/1.11-Unfair-Practice-Procedure.pdf)

**为什么要在作业中复制代码？**

许多大型软件开发涉及重用较小的代码块，然后将其调整为与其他小代码块协同工作。有时这些重用的代码位于库模块中，有时该代码可以在 MDN 等教程网站或 Stack Overflow 等问答网站上找到。

根据您设置的任务，您可能认为应该重用其中一些站点中的代码，以便：加快应用程序的开发、尝试未使用的技术或解决您正在努力解决的问题。

是否应重用代码将取决于赋值集。你不会因为编写复制的代码而获得任何分数（因为你没有），但你会因为将其集成到你的解决方案中获得分数，因此你必须考虑作业的目标和评分标准。

举一个简单的例子来说明这一点：

您的 Java 任务是编写一个 java 程序来对数字数组执行冒泡排序。

您从 Stack Overflow 复制（并引用）一个代码片段，该代码片段对数字数组执行冒泡排序，并仔细地将算法放入所需的方法签名中。

这错过了练习的重点，您没有为算法的生成获得任何分数，并且集成非常少，因此可能不会因此获得任何分数。

但是，如果：

您的商业应用程序任务是创建一个网站，该网站将收集学生成绩数据，存储这些数据，并能够将该格式化数据导出为按成绩排序的 JSON 文件。

从 Stack Overflow 复制（并引用）一个代码片段，该代码片段对数字数组执行冒泡排序，仔细地将算法放入方法签名中，然后将此方法集成到您的项目中。

你不会因为对冒泡排序进行编码而获得分数，但你会因为将该代码集成到一个更大的项目中而获得分数。您将获得决定使用冒泡排序的分数（尽管可能会为不同的方法提供更多分数）。总而言之，除非作业另有规定，否则如果您自己写，您的分数可能会几乎没有差异。

**如何在代码中引用代码**

就像当你在写作中包含其他人的想法和文字时，你必须将它们归因于作者一样，当你在你的代码中包含其他人的代码时，你也必须这样做;否则，你就是在假装这是你自己的作品。

如果你的讲师没有明确定义他们希望你如何引用任何复制的代码，下面给出了一些一般的例子。

在代码中引用代码并不需要太多。如果您复制了一段代码，则可以在注释中引用它，靠近包含复制代码的位置。您至少应该包含一些有关复制代码的一般信息、从中复制代码的 URL 以及访问日期，例如

用于执行冒泡排序的代码

摘自 A.C Oder 20-6-2004 的 Stack Overflow 帖子

访问时间：2018年4月9日

[https://stackoverflow.com/questions/。 /bubblesort-实现](https://stackoverflow.com/questions/........./bubblesort-implementation)

[[ 代码在这里 ]]

引用代码的结尾。

或

您可以将此信息包含在方法或函数的文档注释中，例如

/\*\*

* 返回排序后的数组
* 此代码摘自 A.C Oder 20-6-2004 的 Stack Overflow 帖子
* 访问时间：2018年4月9日
* [https://stackoverflow.com/questions/。 /bubblesort-实现](https://stackoverflow.com/questions/........./bubblesort-implementation)
* [@param](http://www.oracle.com/technetwork/java/javase/documentation/index-137868.html#%40param) 输入未排序数组
* [@return](http://www.oracle.com/technetwork/java/javase/documentation/index-137868.html#%40return) 排序数组

\*/

您可能不得不调整大量复制的代码，以使其适合您的需求。在这种情况下，您可以声明代码已被改编并突出显示您更改的行/块，以便标记可以归功于您的工作。

/\*\*

* 返回排序后的数组
* 此代码改编自 A.C Oder 20-6-2004 的 Stack Overflow 帖子
* 访问时间：2018年4月9日
* [https://stackoverflow.com/questions/。 /bubblesort-实现](https://stackoverflow.com/questions/........./bubblesort-implementation)
* 添加了代码以允许从高到低或从低到高排序
* 查看更多改编评论
* [@param](http://www.oracle.com/technetwork/java/javase/documentation/index-137868.html#%40param) 输入未排序数组
* [@param](http://www.oracle.com/technetwork/java/javase/documentation/index-137868.html#%40param) 真高到低，假低到高
* [@return](http://www.oracle.com/technetwork/java/javase/documentation/index-137868.html#%40return) 排序数组

\*/

然后，在代码中，您应该突出显示您希望通过附加注释进行调整而获得赞誉的部分。

**更多资源：**

麻省理工学院关于在代码中引用的指南： <http://integrity.mit.edu/handbook/writing-code>