**考试说明**

1. **考试时间：合计120分钟，选择题部分不超过30分钟，允许提前交卷。**
2. **考试过程中，不能连接未经指定网站或服务器。**
3. **闭卷考试部分，不能查阅任何类型的参考资料。**
4. **开卷考试部分，可以查阅纸质文档，不能查阅除Python编程环境自带帮助文件以外的任何类型的电子文档。**
5. **考试过程中，不得使用任何形式的电子存储设备，不可使用手机。**
6. **违反上述2-5条者，视为考试作弊。**

**选择题答题方式（20分，闭卷，自动阅卷，严禁使用python编程环境进行尝试）**

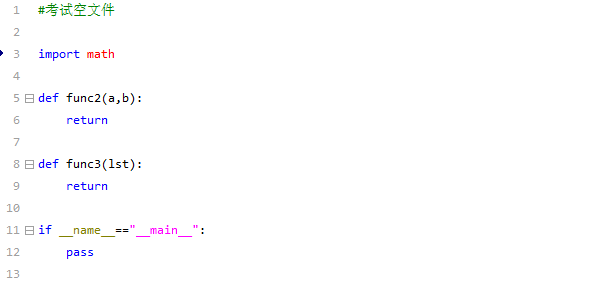
1. **打开浏览器，在地址栏中输入**http://192.168.125.3**，点击相应链接进入登录页面。（未能返校的同学需要首先登录苏州大学VPN服务器）**
2. **按要求输入两遍自己的学号，考场号为所在机房号码。**
3. **点击“登录”按钮即可进入答题页面。如考试尚未开始，系统会进入等待页面并倒计时。考试开始时间到，系统会自动进入答题页面。**
4. **在页面左侧选择题号，页面右侧即会显示相应的题目。考生只需点击选择相应的选项。**
5. **答题过程中如关闭浏览器或出现系统故障导致计算机重新启动，系统不会丢失之前已经完成的题目的答案。考生可以打开浏览器重新登录并继续考试。**
6. **答题完成后，点击“交卷”按钮即可完成交卷。交卷后不能再次登录系统继续考试。**
7. **考试结束时间到，系统会自动收卷。**

**编程题注意事项与提交方式（80分，开卷，自动阅卷）**

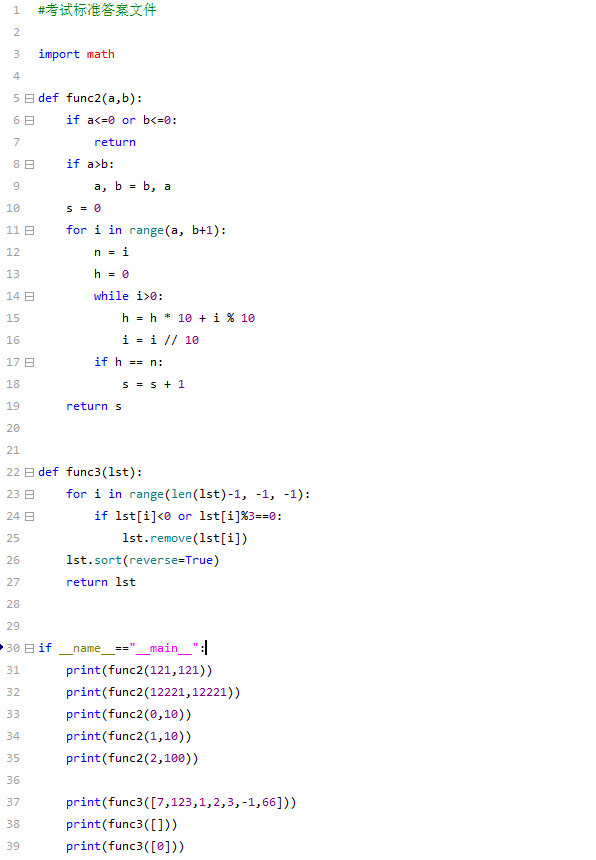
1. **下载sample.py，保存到D盘。严禁将代码文件或数据文件保存在C盘中或桌面上。**
2. **根据题目要求补全sample.py中的空函数。sample.py中的已有函数的函数名、参数数量和顺序不可以修改。**
3. **测试代码，请写入到if \_\_name\_\_=="\_\_main\_\_":部分，不要写到全局环境中。**
4. **不要在代码中调用input()，否则会导致阅卷失败。如阅卷失败，编程题部分不得分。**
5. **每个函数中不需要使用print()输出结果，而是用return返回结果。如用print()输出结果将导致该函数不得分。**
6. **不要使用关键字（if、else、break、def等，会导致语法错误）作为自己的变量名，也不要使用内嵌名字（如list、int、input等，没有语法错误，但是很危险）作为自己的变量名，例如：list = []。**
7. **代码中不要出现任何中文。**
8. **提交的时候，确保程序可以正常运行，不要遗留任何语法错误。语法错误、程序异常等将导致阅卷失败。如阅卷失败，编程题部分不得分。**
9. **所有脚本程序内容必须仅包含在一个脚本程序文件（py文件）中。**
10. **确保文件已经存盘，并知道文件存放的位置。提交前务必关闭IDLE或PyCharm、VS Code编程环境。如文件丢失或无法找到，编程题部分不得分，不能重考。**
11. **考试过程中，请经常进行存盘操作，以防突发异常情况而导致程序未保存。**
12. **在浏览器的地址栏中输入**http://192.168.125.3**，点击相应链接进入提交页面。**
13. **按要求输入两遍自己的学号。**
14. **点击“选择文件”按钮，选择自己的脚本程序文件。点击“提交”按钮提交。**
15. **如提交成功，系统会显示相关信息。如果提交不成功，请重复步骤25-27。**
16. **提交成功后，可点击“查看内容”按钮检查提交的内容。**
17. **编程题部分由阅卷系统自动批阅，以运行正确的测试用例数计分。不采取人工阅卷的方式。**

**答题方式示意图：**

1. **请根据图1、图2和图3所示的说明严格规范源文件结构。**



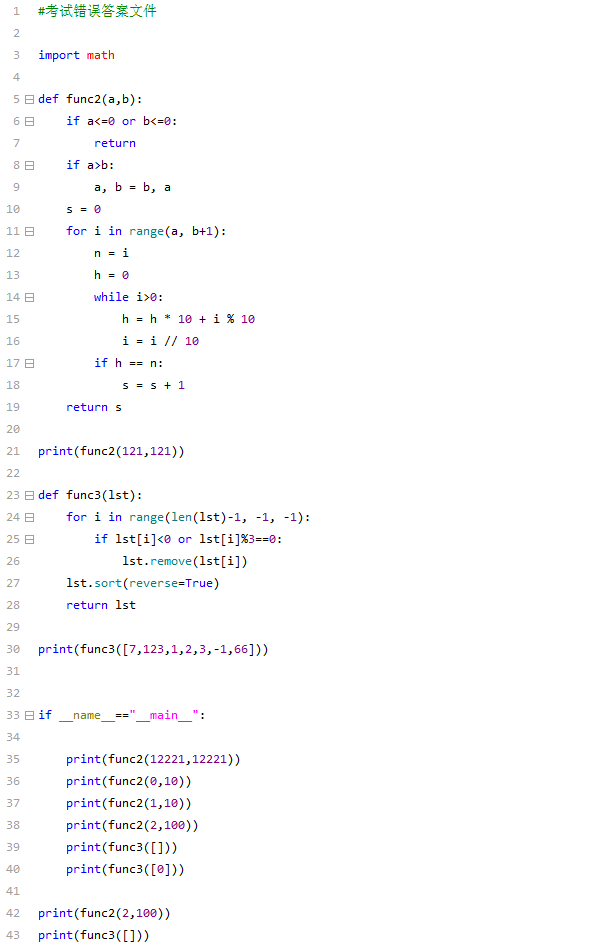
**图1 空文件样例**



测试代码写在此处，且全部包含的if语句体内，即保持相同的缩进位置。

每道题目的代码包含在一个确定名称的函数体内。函数中不包含input函数和print函数。

**图2 正确的答题文件格式**



错误！！！不要在函数之间插入测试代码

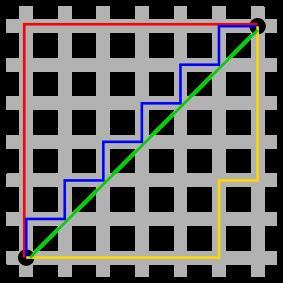
错误！！！不要在函数之间插入测试代码

错误！！！不要在if语句体外插入测试代码

**图3 错误的答题文件格式**

**题目说明**

1. 给定二维平面坐标系上的两个点的坐标[x1,y1]和[x2,y2]，计算两点间的曼哈顿距离。



上图中红线代表曼哈顿距离，绿线代表欧氏距离，也就是直线距离，而蓝线和黄线代表等价的曼哈顿距离。曼哈顿距离——两点在南北方向上的距离加上在东西方向上的距离，即d(i,j)=|xi-xj|+|yi-yj|。对于一个具有正南正北、正东正西方向规则布局的城镇街道，从一点到达另一点的距离正是在南北方向上旅行的距离加上在东西方向上旅行的距离，曼哈顿距离又称为出租车距离。

|  |  |
| --- | --- |
| 相关说明 | |
| 输入条件 | * 表示两点坐标的两个列表 * 列表的0号元素表示x坐标，1号元素表示y坐标 |
| 输出要求 | * 计算两点的曼哈顿距离并返回 |
| 其它要求 | 将代码写入函数func1 |

测试用例：

|  |  |
| --- | --- |
| 输入 | 返回 |
| [1,0],[0,1] | 2 |
| [1,1],[2,2] | 2 |
| [2,2],[2,2] | 0 |
| [1,3],[2,3] | 1 |

1. 给定一元二次方程的三个系数，利用求根公式计算出较大的那个解，如果无解则返回None。

|  |  |
| --- | --- |
| 相关说明 | |
| 输入条件 | * 参数a、b、c都是整数，分别代表一元二次方程的三个系数。 * 方程形式：ax2+bx+c=0 |
| 输出要求 | * 返回较大的解的整数部分。 * 如果无解返回None |
| 其它要求 | 将代码写入函数func2。 |

测试用例：

|  |  |
| --- | --- |
| 输入 | 返回 |
| 1,2,1 | -1 |
| 1,-4,3 | 3 |
| 1,2,3 | None |

1. 某停车场的收费标准如下：(1)30分钟(含30分钟)内免费；(2)(30,60]分钟内收5元；(3)超过60分钟，少于等于240分钟，则超过部分每15分钟收2元，不足15分钟按15分钟计算；(4)超过240分钟，则每30分钟收5元，不足30分钟按30分钟计算；(5) 一天24小时封顶150元；(6) 超过24小时则重新按照(1)~(5)计算。现有某车的停车时间t(单位为分)，请计算该车应该交的停车费。

|  |  |
| --- | --- |
| 相关说明 | |
| 输入条件 | * 表示停车时间的正整数t |
| 输出要求 | * 返回应交停车费 |
| 其它要求 | 将代码写入函数func3 |

测试用例：

|  |  |
| --- | --- |
| 输入 | 返回 |
| 30 | 0 |
| 240 | 29 |
| 300 | 39 |
| 1470 | 150 |
| 1440 | 150 |
| 1500 | 155 |

1. 已知某数列的递推式为：；现给定数列的和,请编写程序计算该数列的第num项的值，这里为数列的第1项，为数列的第2项。

|  |  |
| --- | --- |
| 相关说明 | |
| 输入条件 | * 整数和 * 正整数num |
| 输出要求 | * 返回数列的第num项的值 |
| 其它要求 | 将代码写入函数func4 |

测试用例：

|  |  |
| --- | --- |
| 输入 | 返回 |
| 0,1,5 | 1 |
| 1,4,10 | 11 |
| 1,4,1 | 4 |
| 1,1,10 | 5 |

1. 给定一个正整数列表lst。将相邻两数之和最大的两个数相加整除2的结果替代这两个数。重复上述过程，直至列表中只剩下2个数。如出现多对相邻数之和相等，则取元素下标小的数对。例如：列表[7,0,7,8]的变化过程为[7,0,7,8] >> [7,0,7] >> [3,7]。

|  |  |
| --- | --- |
| 相关说明 | |
| 输入条件 | * 参数lst是正整数列表。 * 所有元素均正整数，无需额外的检查。 |
| 输出要求 | * lst中所有不同的正奇数的增序序列。 * 如果输入的列表长度小于2，则返回空列表。 |
| 其它要求 | 将代码写入函数func5 |

测试用例：

|  |  |
| --- | --- |
| 输入 | 返回 |
| [3,0,7,8] | [3, 3] |
| [1,9,7,4,3] | [1, 4] |
| [2] | [] |

1. 给定一个整数列表lst。找出其中非空（最少有1个元素）递增子序列的个数。后一个元素不小于前一个元素即为递增。

|  |  |
| --- | --- |
| 相关说明 | |
| 输入条件 | * 参数lst是整数列表。 * 所有元素均为整数，无需额外的检查。 |
| 输出要求 | * 返回一个整数。 * 如果lst为空，返回-1。 |
| 其它要求 | 将代码写入函数func6 |

测试用例：

|  |  |
| --- | --- |
| 输入 | 返回 |
| [28, 38, 48, 68] | 1 |
| [28, 58, 38] | 2 |
| [12, 12, 4, 126, 8, 12] | 3 |
| [1, 9, 7, 4, 3] | 3 |

1. 给定一个由正整数组成的一维列表lst，如果列表中偶数的个数等于奇数的个数，则返回1；如果偶数的个数大于奇数的个数，则返回2；偶数的个数小于奇数的个数，则返回3。

|  |  |
| --- | --- |
| 相关说明 | |
| 输入条件 | * 一维列表lst只包含正整数 * 列表长度大于等于1 |
| 输出要求 | * 个数相等，返回1 * 偶数个数大于奇数个数，返回2 * 偶数个数小于奇数个数，返回3 |
| 其它要求 | 将代码写入函数func7 |

测试用例：

|  |  |
| --- | --- |
| 输入 | 返回 |
| [8, 9] | 1 |
| [8, 8] | 2 |
| [9, 9] | 3 |

1. 某次考试以后，用一个二维列表lst存储了考试结果，其中lst的每个小列表存储的信息如下：[学号，数学分数，语文分数]。根据考试总分进行降序排序，如果总分相同则根据学号进行升序排序，最后以二维列表的形式返回总分最高(含并列)的同学和对应的总分。举例：给定lst=[[2001, 90, 90], [2002, 70, 70], [2003, 80,100], [2004,60,60]]，最后返回[[2001, 180], [2003, 180]]

|  |  |
| --- | --- |
| 相关说明 | |
| 输入条件 | * 列表lst的长度大于等于1 * 学号、考试分数都是正整数 |
| 输出要求 | * 以二维列表形式返回结果 |
| 其它要求 | 将代码写入函数func8。 |

测试用例：

|  |  |
| --- | --- |
| 输入 | 返回 |
| [[2002, 100, 100], [2001, 90,80]] | [[2002, 200]] |
| [[2001, 90, 90], [2002, 95,95]] | [[2002, 190]] |
| [[2002, 80, 90], [2001, 90,80]] | [[2001, 170], [2002, 170]] |