**Python实验一报告**

**实验题目：设计一个凯撒密码（Caesar cipher）加密和解密程序**

**专 业：人工智能**

**姓 名：李洋**

**学 号：22920192203994**

**实验日期：2021/3/25**

1. **实验目的**

设计实现针对中文的凯撒密码的加密和解密程序。

1. **实验步骤以及结果**

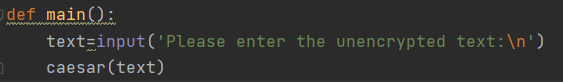
**1.加密程序**

**（1）数据描述：**

数据为一段仅由汉字组成的字符串，以回车结尾

**（2）方法描述：**

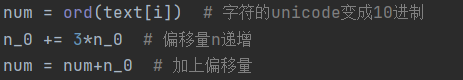
1. 将输入的明文保存在字符串text中，然后传入caesar函数进行加密



1. 使用随机数确定偏移量n，但是偏移量的随机数的上限有要求，为了准确传递偏移量，n的初始值的最大值应该是汉字范围大小（20901）的开根；然后将初始偏移量n赋值给n\_0（具体原因后续步骤会进行讲解）



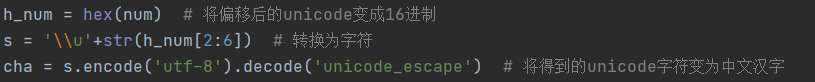
1. 遍历字符串，对每一个字符进行加密。首先得到字符的10进制的unicode编码，然后加上偏移量n\_0，注意，随着每一个字符的加密偏移量n其实是递增的



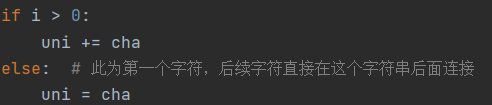
1. 然后判断加上偏移量n后字符的unicode编码是否超出了汉字的范围[19968,40869]，则取模范围长度，并加上最小值



1. 将新获得的unicode编码转变为汉字



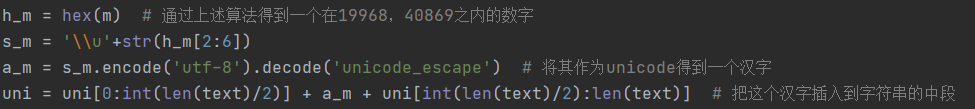
1. 如果是第一个字符，则作为一个新字符串uni的开头，如果是后续字符则直接连接在新字符串uni的后面



1. 遍历完字符串后，对初始偏移量n进行加密，先平方再加上19968（汉字的最小unicode）和163（小于（20901-144\*144）=165的最大质数）。我为了初始偏移量能更准确的加密，让n的平方在汉字的范围长度之内即n^2属于[0,20901]，所以经过计算得到n的最大值不超过144.但144的平方实际上与20901还相差165，所以我选择了比165小的最大质数163，然后再加上汉字的最小unicode=19968，这样就能确保无论n为多少，计算得到的m一定在汉字的范围内



1. 同第六部操作，将m转换为汉字字符，并将其插入在新字符串uni的中间位置



1. 然后输出这个新字符串uni，即加密后的暗文

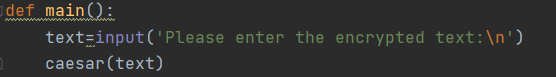


1. **解密部分**
2. **数据描述：**

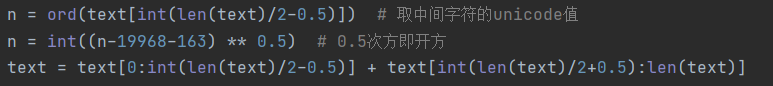
数据为一段仅由汉字组成的字符串，以回车结尾

**（2）方法描述：**

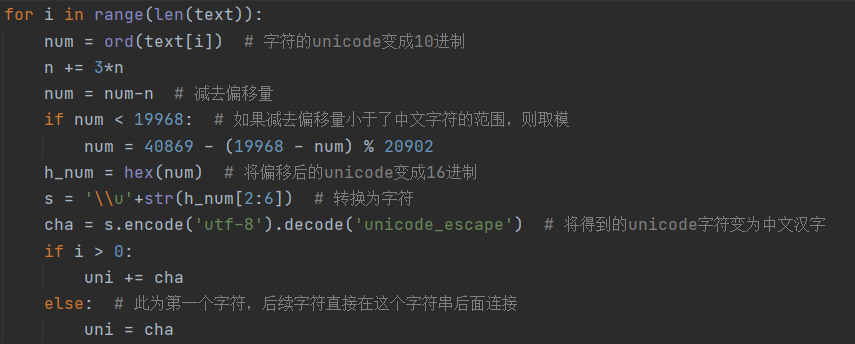
1. 将输入的暗文保存在字符串text中，然后传入caesar函数进行解密

****

1. 取出text的中间字符（并在text中删除），将其通过偏移量加密函数的反函数得到初始偏移量n



1. 与加密的步骤相类似，先算出对应字符的偏移量n，然后减去，如果新的unicode编码超出范围则进行取模运算，然后再由新的编码转换为新的字符，再逐个拼接成新的字符串uni

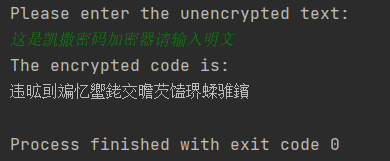


1. 最后输出uni



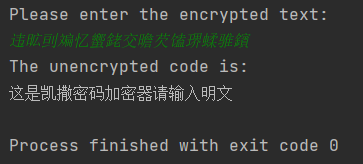
1. **实验结果与分析**
2. **第一次试验**
3. 加密明文：这是凯撒密码加密器请输入明文

得到暗文：违昿刯斒忆蠁銠交曕芡馌琾蝚骓鑌

****

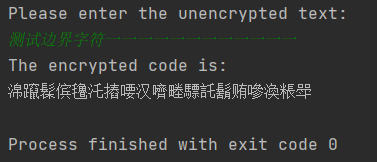
1. 解密暗文：违昿刯斒忆蠁銠交曕芡馌琾蝚骓鑌

得到明文：这是凯撒密码加密器请输入明文



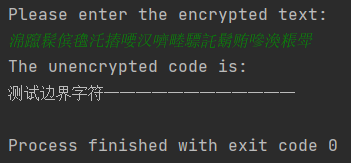
1. **第二次试验**
2. 加密明文：测试边界字符一一一一一一一一一一一一

得到暗文：淿躥髹傧氌汑摏喓汉嚌畻驃託鬍贿嘇渙粻斝



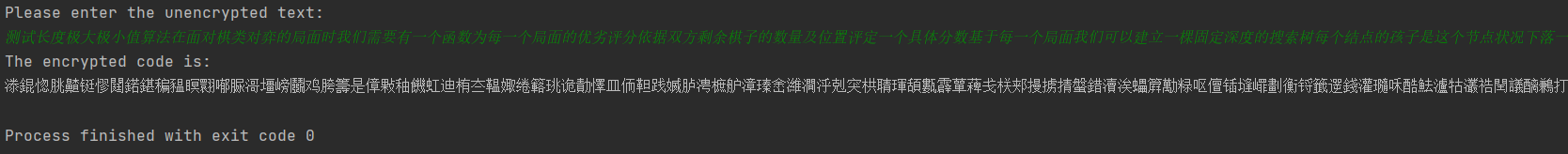
1. 解密暗文：淿躥髹傧氌汑摏喓汉嚌畻驃託鬍贿嘇渙粻斝

得到明文：测试边界字符一一一一一一一一一一一一



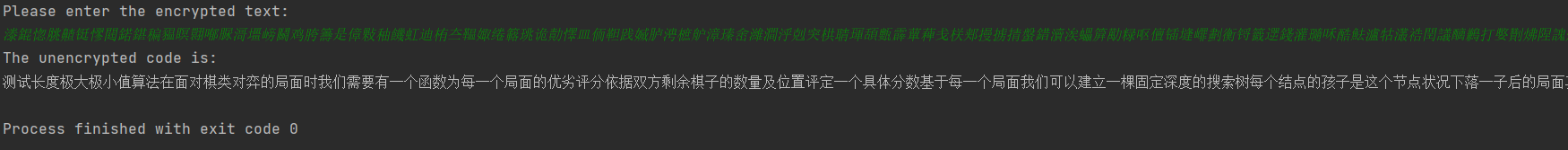
1. **第三次试验**
2. 加密明文：测试长度极大极小值算法在面对棋类对弈的局面时我们需要有一个函数为每一个局面的优劣评分依据双方剩余棋子的数量及位置评定一个具体分数基于每一个局面我们可以建立一棵固定深度的搜索树每个结点的孩子是这个节点状况下落一子后的局面其叶子节点不必是棋局结束的最终状态可以只是固定深度的最深一层节点每个结点拥有一个由上述函数评出的值这个函数就是评估函数这样就构建了一个可以知晓所有可能局面以及这个局面是否有利的搜索树在这个搜索树中，我们假设电脑先手那么第一层就是电脑的所有可能的走法第二层就是玩家的所有可能走法以此类推选择走哪一个分支则需要一个评估函数来对当前整个局势作出评估这个节点的评估函数值越大则进行这个结点的分支的走法对电脑越有利而对于极大极小值算法来说我们规定电脑走子的层为最大层这一层中为了保证电脑自己利益的最大化需选择评估值最高的结点而玩家走子的层为最小层这一层中为了保证玩家自己的利益最大化就会选择评估值最低的结点也就是说电脑以对手每次都走最有可能导致我方失败的走法为前提进行决策找出对自己最有利的走法争取最小化对手可能的最大利益限制对手的上限依据这一规则计算机就会计算出下一步的走法这一步的走法将是所有走法里最不可能让玩家获胜的走法

得到暗文：漛錕惚脁齄铤憀閮鍩鍖稨豱瞑翾喐脲滒壃嵭鬭鸡胯籌是傽敤秞饑虹迪栯夳鞰娵绻簵珧诡勣懌皿侕靼践媙胪涄樜舮漳瑧峹濰澗泘剋突栱聙琿頢甊霹蕇薭戋栚郏摱掳掅螌錯瀆涘蠝簈勱粶呕儃锸塳嶵劃衡锊籤遻錢灌瓍咊酷魼瀘牯灇祰閏議醨鶼打嫳劗炥陧謉熵草畍飍砼氖虀峌铢楍慞铔褉彫恱鲣摦斀锦矹匜屑烳熬涏恿悏吪設堁煻繀偂茿帲酾鑐襦藋訇澰覩襷覼佷薃誙搋埻崬硥垚篡摌蚘標母狈雳惰俜鸎稃坍腵癭鸵鼦蛌抌抒匹粐塢鋆冻测魸鄡遀促抡弽窊叩昒乊朅矃稕寚矯澦炧鱧楙檐堦莡幆酣攮趱糾二簣橨彮繣顉烋闅黮簮箈驚觿広堒榶虰豍鋫芾脧晵樽鐚僞瀈黕踋溳飒矄洂輧玮橆蜿騾褋艆綀命昍銉歙甙茳丬饖浙蘮鉗檑娑絓机梻璎邚躈毒租中屉勠保叻鏅漱沎摇襍询酿呡暓巠铹欑喏渪呼饤峇蔎璡畠鲐芑舷芤骯竍珬硒铰岮齇屔磸矹赐炛鸌逄一饔偛簐溶滦塏衬隯戵悧襦耬觙麗身欵訃愯鎆僉懲譓岙來厛搇吸鴪椰僺擗賮櫎螵梇肯渺喹俣梄栞仯僋嚥股帍硇藜賚黧鵬彳鱡霑炓阡鉧芿祢翥笖鸽嚛褴震糅斊郌蟒媱緓鼸蚫鋂萹鲹貼縛擶爫磾塠憝鮃貓捧銍躋赆壜鈴皐縂馗娘晀蛪颾恪摈麨樤镕蚑鎤挌毈壋甭肘踬垮蜖幅縗脔鼳侖壱摜継碊獣鑕瘇骄蜡狊梻淲淊鍟嶍傒胆祻婅竆籉嵶格殰蹘婇饕缬颔鮁藝戍魒檺頽荅癇狂鹙蚐穿顱賊腰勐杧峢凛隢嗲庯胜痵柪梕喱苈闼暉篠曇乹旨釵骜邥絎铛詾攡垝戤劢篆益燐呏昘尃虩筛瘤



1. 解密暗文：漛錕惚脁齄铤憀閮鍩鍖稨豱瞑翾喐脲滒壃嵭鬭鸡胯籌是傽敤秞饑虹迪栯夳鞰娵绻簵珧诡勣懌皿侕靼践媙胪涄樜舮漳瑧峹濰澗泘剋突栱聙琿頢甊霹蕇薭戋栚郏摱掳掅螌錯瀆涘蠝簈勱粶呕儃锸塳嶵劃衡锊籤遻錢灌瓍咊酷魼瀘牯灇祰閏議醨鶼打嫳劗炥陧謉熵草畍飍砼氖虀峌铢楍慞铔褉彫恱鲣摦斀锦矹匜屑烳熬涏恿悏吪設堁煻繀偂茿帲酾鑐襦藋訇澰覩襷覼佷薃誙搋埻崬硥垚篡摌蚘標母狈雳惰俜鸎稃坍腵癭鸵鼦蛌抌抒匹粐塢鋆冻测魸鄡遀促抡弽窊叩昒乊朅矃稕寚矯澦炧鱧楙檐堦莡幆酣攮趱糾二簣橨彮繣顉烋闅黮簮箈驚觿広堒榶虰豍鋫芾脧晵樽鐚僞瀈黕踋溳飒矄洂輧玮橆蜿騾褋艆綀命昍銉歙甙茳丬饖浙蘮鉗檑娑絓机梻璎邚躈毒租中屉勠保叻鏅漱沎摇襍询酿呡暓巠铹欑喏渪呼饤峇蔎璡畠鲐芑舷芤骯竍珬硒铰岮齇屔磸矹赐炛鸌逄一饔偛簐溶滦塏衬隯戵悧襦耬觙麗身欵訃愯鎆僉懲譓岙來厛搇吸鴪椰僺擗賮櫎螵梇肯渺喹俣梄栞仯僋嚥股帍硇藜賚黧鵬彳鱡霑炓阡鉧芿祢翥笖鸽嚛褴震糅斊郌蟒媱緓鼸蚫鋂萹鲹貼縛擶爫磾塠憝鮃貓捧銍躋赆壜鈴皐縂馗娘晀蛪颾恪摈麨樤镕蚑鎤挌毈壋甭肘踬垮蜖幅縗脔鼳侖壱摜継碊獣鑕瘇骄蜡狊梻淲淊鍟嶍傒胆祻婅竆籉嵶格殰蹘婇饕缬颔鮁藝戍魒檺頽荅癇狂鹙蚐穿顱賊腰勐杧峢凛隢嗲庯胜痵柪梕喱苈闼暉篠曇乹旨釵骜邥絎铛詾攡垝戤劢篆益燐呏昘尃虩筛瘤

得到明文：测试长度极大极小值算法在面对棋类对弈的局面时我们需要有一个函数为每一个局面的优劣评分依据双方剩余棋子的数量及位置评定一个具体分数基于每一个局面我们可以建立一棵固定深度的搜索树每个结点的孩子是这个节点状况下落一子后的局面其叶子节点不必是棋局结束的最终状态可以只是固定深度的最深一层节点每个结点拥有一个由上述函数评出的值这个函数就是评估函数这样就构建了一个可以知晓所有可能局面以及这个局面是否有利的搜索树在这个搜索树中寀我们假设电脑先手那么第一层就是电脑的所有可能的走法第二层就是玩家的所有可能走法以此类推选择走哪一个分支则需要一个评估函数来对当前整个局势作出评估这个节点的评估函数值越大则进行这个结点的分支的走法对电脑越有利而对于极大极小值算法来说我们规定电脑走子的层为最大层这一层中为了保证电脑自己利益的最大化需选择评估值最高的结点而玩家走子的层为最小层这一层中为了保证玩家自己的利益最大化就会选择评估值最低的结点也就是说电脑以对手每次都走最有可能导致我方失败的走法为前提进行决策找出对自己最有利的走法争取最小化对手可能的最大利益限制对手的上限依据这一规则计算机就会计算出下一步的走法这一步的走法将是所有走法里最不可能让玩家获胜的走法



注：1.由于加密操作是由随机数进行的，重复明文并不一定能得到同样的暗文

2.第三次实验的运行结果截图并不完整，但是已经将两次运行的结果以文本的形式写了出来

1. **实验总结**

这次的实验是我第一次尝试用PyCharm编写python源文件并运行，结合老师课上教的知识和我在网上查阅到资料，我完成了此次编写。

此次实验的难点在于如果我选择用随机数作为初始偏移量，那么如何只通过暗文来传递信息又不被轻易截获，所以我想到了将信息加密后以与其他字符相类似的形式隐藏在暗文中，然后解码时就能通过这个信息字符得到加密时的初始偏移量来完成解码。

总结来说这个加密程序有三道“工序”：

1）取随机数作为偏移量

2）偏移量会随着加密字符的次数递增（但是能确保偏移后的字符在给定的汉字范围）

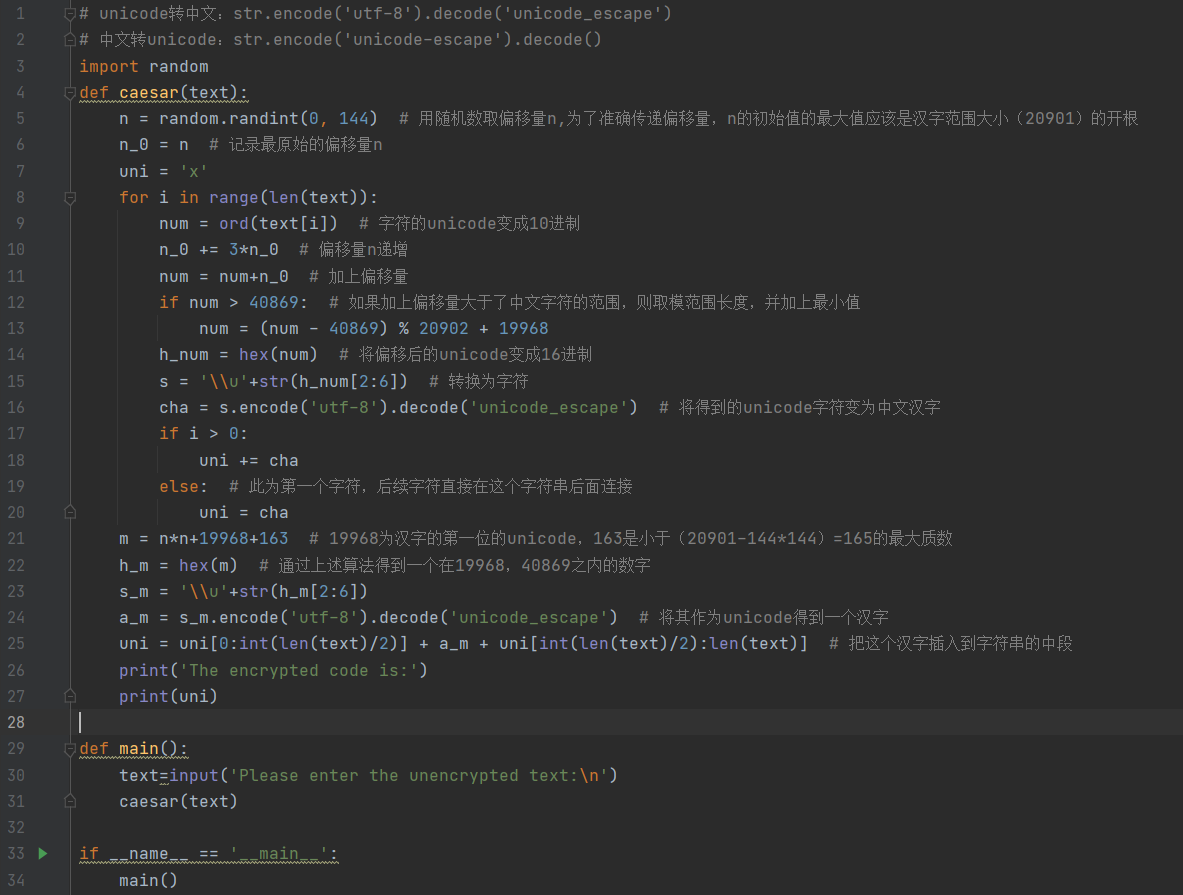
3）用非线性函数加密初始偏移量，并将其伪装成字符藏在字符串中（其实即使破解者发现了这个信息字符，他也不知道我的加密函数，无法通过信息字符得到偏移量；更何况此程序的偏移量是随机的，每一次都在变化，暴力破解也几乎不可能推断出中间字符为信息字符）

最后，再提一下写代码时犯的一些错：函数和部分控制语句后面缺少冒号，使用random函数时为加入头文件，不规范的缩进等，在多次的优化和改进后我终于完成了此次实验。

所以，通过此次实验，我的收获有：随机函数的使用、unicode编码和汉字字符之间的转换、通过非线性函数来降低暴力破解的可能性，以及部分控制语句的格式规则等，我感觉收获颇丰，希望能将它们再次运用到后续的python学习之中。

附件：

1.加密器源代码



2.解码器源代码

