**奶茶店经营模拟游戏说明文档**

3220104465 徐文钧

**一、游戏设计思想**

本游戏旨在通过模拟奶茶店经营的场景，让玩家在游戏过程中学习和应用经济学原理，如供需关系、成本与利润、价格弹性等，培养玩家的经济决策能力和经营策略思维。游戏以阶段式经营为核心玩法，玩家需要在不同市场环境下做出合理的制作量和售价决策，以实现利润最大化。

**二、游戏背景及经济学原理应用**

游戏背景设定为玩家经营一家奶茶店，面对四季变化和春节等不同阶段的市场环境，玩家需要考虑以下经济学原理：

1. 供需关系：售价越高，需求越低，玩家需要找到价格与需求的平衡点，以实现最大利润。

2. 库存管理：库存过多会导致浪费，库存过少会错失销量，玩家需要合理安排制作量。

3. 成本与利润：控制成本，保持合理售价，才能实现利润最大化。

4. 价格弹性：不同季节顾客对价格的敏感度不同，夏季价格敏感度更高，玩家需要根据季节调整售价策略。

5. 风险管理：市场波动不可预测，玩家需要保持应急资金以应对突发情况。

**三、核心玩法及机制说明**

**1. 游戏目标**

在有限的经营阶段内，通过合理决策，积累尽可能多的资金。

**2. 游戏流程**

游戏开始时，玩家有初始资金10000元，库存为0，售价默认为20元。

每个阶段，玩家需要输入奶茶制作数量（1500-3000杯）和售价（10-30元），然后点击“确认本阶段经营”。

游戏根据当前阶段的市场环境（如季节、顾客基数、售价系数、成本系数等）计算实际需求量，并根据玩家的制作量和售价计算成本支出、收入和利润。

游戏会在每个阶段结束后显示经营总结，并提示相关的经济学知识点。

游戏共5个阶段，结束后会显示最终资金余额，并进入公共经济学问答环节，答对题目可获得额外奖励。

**3. 经营决策影响**

（1）制作量决策：制作量过多可能导致库存积压浪费成本，制作量过少可能导致无法满足需求而错失利润。

（2）售价决策：售价过高会降低需求量，售价过低会减少利润空间，玩家需要根据市场环境和成本因素综合考虑定价策略。

**四、如何运行和操作游戏**

**1. 运行环境**：确保安装了Python环境，并安装了`tkinter`库（通常Python自带`tkinter`库）。

**2. 运行步骤**：

将`GAME.py`文件保存到本地。

打开终端或命令提示符，导航到文件所在目录。

输入命令`python GAME.py`运行游戏。

**3. 游戏操作**：

游戏开始时，点击“开始游戏”按钮进入游戏界面。

在游戏界面中，输入奶茶制作数量和售价，然后点击“确认本阶段经营”按钮，开始本阶段的经营。

游戏会根据玩家的决策计算经营结果，并在日志区域显示经营总结和相关经济学知识点。

游戏结束后，会进入公共经济学问答环节，答对题目可获得额外奖励。

**五、代码结构说明**

**1. 模块划分**

（1）全局变量和常量：

定义了游戏相关的全局变量和常量，如最大阶段数、制作量范围、售价范围、阶段环境参数、每日经济学知识点、公共经济学问答题目等。

（2）游戏类`MilkTeaShopGame`：

`\_\_init\_\_`方法：初始化游戏窗口、初始资金、库存、阶段、售价等属性，并创建开始界面。

`create\_start\_screen`方法：创建游戏开始界面，显示游戏介绍和开始按钮。

`show\_knowledge\_point`方法：显示经济学知识点总结，包括开始前知识点和每个阶段经营总结。

`start\_game`方法：开始游戏，重置游戏状态，创建游戏界面。

`create\_game\_screen`方法：创建游戏界面，包括状态显示区域、输入区域、需求函数显示区域和经营日志区域。

`update\_stage\_info`方法：更新当前阶段的市场环境信息和需求函数显示。

`play\_stage`方法：处理玩家的经营决策，计算成本支出、实际需求量、销售量、收入和利润，并更新游戏状态和日志。

`append\_log`方法：将经营总结信息追加到日志区域。

`end\_game`方法：结束游戏，显示最终资金余额，并进入公共经济学问答环节。

`show\_public\_econ\_quiz`方法：显示公共经济学问答题目，根据玩家答题情况给予奖励。

`clear\_screen`方法：清除当前窗口中的所有组件，用于切换界面。

1. **关键代码段功能解释**

（1） `play\_stage`方法：

```python

def play\_stage(self):

try:

make\_count = int(self.make\_entry.get())

price = float(self.price\_entry.get())

except ValueError:

messagebox.showerror("输入错误", "请输入有效的数字。")

return

if not (MIN\_MAKE <= make\_count <= MAX\_MAKE):

messagebox.showerror("输入错误", f"制作数量应在{MIN\_MAKE}至{MAX\_MAKE}之间。")

return

if not (MIN\_PRICE <= price <= MAX\_PRICE):

messagebox.showerror("输入错误", f"售价应在{MIN\_PRICE}至{MAX\_PRICE}元之间。")

return

env = STAGE\_ENVIRONMENTS[self.stage 1]

cost\_per\_cup = self.cost\_per\_cup\_base env["cost\_factor"]

total\_cost = cost\_per\_cup make\_count

if total\_cost > self.money:

messagebox.showerror("资金不足", f"资金不足支付成本：需要{total\_cost:.2f}元，当前资金{self.money:.2f}元。")

return

self.price = price

self.stock = make\_count

self.money = total\_cost

base\_demand = env["customer\_base"] env["price\_coef"] price

fluctuation = random.uniform(0.95, 1.05)

demand = max(0, int(base\_demand fluctuation))

sales = min(demand, self.stock)

revenue = sales price

self.money += revenue

self.stock = sales

self.append\_log(f"阶段 {self.stage} 经营总结：")

self.append\_log(f"制作奶茶：{make\_count} 杯，售价：{price:.2f} 元/杯")

self.append\_log(f"预估需求量（含波动）：{demand} 杯，实际售出：{sales} 杯")

self.append\_log(f"成本支出：{total\_cost:.2f} 元，收入：{revenue:.2f} 元")

self.append\_log(f"库存剩余：{self.stock} 杯，资金余额：{self.money:.2f} 元\n")

self.show\_knowledge\_point(self.stage 1)

self.stage += 1

if self.stage > MAX\_STAGES:

self.end\_game()

else:

self.update\_stage\_info()

self.make\_entry.delete(0, tk.END)

self.make\_entry.insert(0, str(MIN\_MAKE))

self.price\_entry.delete(0, tk.END)

self.price\_entry.insert(0, str(self.price))

```

该方法是游戏的核心逻辑部分，主要功能如下：

* 获取玩家输入的制作数量和售价，并进行有效性检查。
* 根据当前阶段的市场环境计算成本支出、实际需求量、销售量、收入和利润。
* 更新游戏状态，包括资金、库存、阶段等信息。
* 将经营总结信息追加到日志区域。
* 显示经济学知识点总结，并根据当前阶段决定是否进入下一阶段或结束游戏。

（2）`update\_stage\_info`方法：

```python

def update\_stage\_info(self):

env = STAGE\_ENVIRONMENTS[self.stage 1]

self.stage\_label.config(text=f"阶段 {self.stage} / {MAX\_STAGES}")

self.money\_label.config(text=f"资金：{self.money:.2f} 元")

self.stock\_label.config(text=f"库存：{self.stock} 杯")

self.price\_label.config(text=f"当前售价：{self.price} 元/杯")

formula\_text = f"需求量 = {env['customer\_base']} {env['price\_coef']} × 价格（元）"

self.demand\_label.config(text=formula\_text)

self.demand\_desc\_label.config(text=f"市场环境描述:\n{env['desc']}")

```

该方法用于更新当前阶段的市场环境信息和需求函数显示，包括阶段标签、资金标签、库存标签、售价标签、需求函数公式和市场环境描述。

1. **使用的外部库**

（1）`tkinter`库：用于创建游戏的图形用户界面，包括窗口、按钮、标签、文本框、消息框等组件，实现游戏的交互功能。

（2）`random`库：用于生成随机数，模拟市场波动，使游戏更具真实性和挑战性。

**六、总结**

本游戏通过模拟奶茶店经营的场景，将经济学原理与游戏玩法相结合，让玩家在轻松愉快的游戏过程中学习和应用经济学知识，培养经济决策能力和经营策略思维。游戏的实现采用了面向对象的编程思想，代码结构清晰，模块划分合理，易于维护和扩展。