**首阶段目标：初步实现题目要求**

1. 总目的
   1. 公司存活时间足够久
   2. 赚取足够的钱
   3. 若a)成立（无限久），b)必然成立；若a)不成立，则应该使得利润最大化，或者思考增加存活时间的策略
2. 动态地图与全局参数
   1. 地图为全局变量
   2. 首阶段为17\*17的地图（后续阶段可以增加）
   3. 现在时间与单位时间
   4. 总金钱
   5. 骑手初始位置（新招募的骑手都在初始位置吗？）
3. 骑手相关（只可往后接续的链表）
   1. 骑手活动参数（骑手活动需要刷新地图）
      1. 骑手方向
      2. 骑手坐标
      3. 已接受的订单（按照派送顺序排序）
      4. 正在派送的订单或欲接收的订单
      5. 到达目的地的最短路程以及时间
   2. 所接的每个订单，订单参数需要完整
   3. 招聘骑手的策略？？？？？（或为难点，首阶段可以对骑手数量使用贪心策略而非对收入使用贪心，因为后者复杂的多）
4. 订单序列（链表）
   1. 默认按时间顺序
   2. 新增订单需要刷新地图（注意一个地方上个订单未完成，就产生下一个订单的特殊情况）
   3. 每个订单都应有自己的状态
      1. 已接受（订单总量）
      2. 未完成（参数1：距离超时的时间限制，参数2：距离破产的时间限制）
      3. 已完成（参数1：是否超时，参数2：获取的利益（可正可负））
      4. 已超时（参数：距离破产的时间限制）
   4. 每个订单，记录下单时间，并以此实时更新超时时间和破产时间
   5. 调整序列，并为每个订单指派一个骑手（按照时间成本最低的原则）
5. 订单策略（非代码实现部分，第一阶段可以酌情实现）
   1. 每个订单，都应该储存对于每一个骑手来说所花的时间成本 ，注意，在储存此个参数前，应当先刷新骑手位置和骑手所接订单

// 如何量化时间成本，在时间成本达到一定值后，使得故意让订单排到序列的后面（甚至超时）的收益高于完成该订单？

* 1. 订单发起后，应该什么时候接收订单（3个时间单位内能做什么）使得收益更高？

1. 判断模块
   1. 超时的判断
   2. 破产的判断
      1. 超时过长
      2. 钱数为负
      3. 拒单
   3. 到达餐馆/食客位置的判断
2. 输入输出（第一阶段可以简单处理）
   1. 输入：控制台输入
   2. 输出：文件输出

**第二阶段目标（粗略）：**

* + - 1. 完善输入输出（文本输入与鼠标输入；文本输出与用户界面输出）
      2. 尝试更大的地图（170\*170）
      3. 优化策略
         1. 招聘骑手的策略

大前提：骑手也属于资产，即总收入的一部分，理论上越多越好

矛盾点1：一方面，招聘骑手可能使得没有钱承受罚款，需要预测估计如果有某种数量或者距离的未完成订单，使得招聘骑手可能导致直接破产，并评估风险（即为了招聘骑手冒这样的风险是否值得，或为概率论知识？）；另一方面，招聘骑手可以使得长远收益更高（能接受更繁重的订单）

矛盾点2：理论上，现金储备足够多时，可以防止招聘骑手而导致没有钱承受罚款；但是依据贪心策略，多出来的储备钱应当用于招聘更多的骑手而非防范风险才能使得收益最大

* + - * 1. 派单的策略

对地图分区，并且指派不同骑手管辖，注意预留跨区域骑手以防意外（题目建议策略）

前期骑手少，每个分区的骑手所分的区域较大，因此跨区域骑手应该较多，防止超出预期数量的跨区域订单（或骑手未达到一定数量时不采用分区策略？）

骑手较多时，跨区域骑手可以缩减到1（不能为0）的前提是，每个骑手负责的区域可以重叠

先近后远还是先远后近

时间不足多少时使得该订单优先？

* + - * 1. 量化时间成本（可能较为困难，可以后期考虑）