第一步，启动separator协议，初始化创建dpnet，一个子网组，内部包含：1个Leader节点，3个follower节点(需自行完成，按照dper的backend\_test.go)

第二步，4个节点各自创建ConsensusPromoter对象，仿照原测试函数过程完成对每个ConsensusPromoter对象的配置(SetLowerChannel)，然后运行ConsensusPromoter对象(Start方法)

第三步，NewCtpbft()创建C-TPBFT协议，绑定到每个ConsensusPromoter对象

第四步，由某一个节点自定义交易集合[]\*eles.Transaction，将这些交易打包为separator协议消息广播发送(其他节点收到此广播的交易消息后会将其添加到对应交易池中—也是ConsensusPromoter对象的功能)—需自行设计

第五步，由于我们已经Start()了ConsensusPromoter对象，因此当newRoundTicker计时器到达计时周期时会自动开启新一轮round(为所有注册的TPBFT)，Leader节点在新一轮round中在交易池中检索未超时的交易消息，将其打包为pre-prepare阶段消息广播给其他follower节点(同时Leader针对这些发送的交易创建RoundHandler处理器)

第六步，其他follower节点的ConsensusPromoter对象在lowerUpdate()检测到pre-prepare消息，会验证其有效性。由于第五步只是Leader节点创建了RoundHandler处理器，follower还没有，因此需要创建相同roundID的RoundHandler处理器，每个follower节点还要广播发送prepare消息，然后启动自身的RoundHandler处理器

第七步，由于在第六步中我们用让每个follower节点又用协程单独发送了prepare消息，相当于又接收到了共识消息msg.IsLowerConsensus()。这一次由于所有节点都有相同ID的RoundHandler处理器，就会将收到的prepare消息解包并存入prepare消息池，然后反复调用roundHandler处理器的Do方法完成prepare阶段、tidy阶段、commit阶段，最后完成组装区块的提交--（完成SubmitBlock）