1. 简述NavMesh的设置方法

答：

对场景中的物体进行标记，然后进行路径烘焙，产生网格数据

为要进行寻路的物体添加寻路组件（NavMeshAgent）

通过NavMeshAgent组件进行移动

1. 简述NavMeshAgent属性参数的使用方法。

答：

destination属性：可以设置目标点

nextPosition:获取或设置模拟navmesh代理的位置，可以手动的驱动角色进去移动

remainingDistance：物体当前位置与目标点的剩余距离。

UpdatePosition，UpdateRotation两个值都为bool值，可以通过更改他们，停止使用Agent组件来驱动物体移动和旋转。

isOnNavMesh： 判断物体是否在导航网格上。

isOnOffMeshLink： 判断物体是否在分离网格链接上。

1. 简述寻路过程中路网烘焙过程

答：

在window中打开Navigation窗口准备烘焙地形

在inspector界面中将场景中的障碍物勾选为静态

在Navigation窗口中进行导航网个的烘焙

设置完成后，直接点击Bake场景中会生成导航网格

生成导航网格后，在需要自动导航的物体上添加NavMeshAgent组件

使用NavMeshAgent类中的函数SetDestination(Vector3 Target);函数来指定导航的终点就可以了

1. 对于寻路过程中的障碍物绕行应该怎样处理?

答：

重新计算路径。我们希望游戏世界随着时间改变，一条一段时间之前发现的路径，可能不再是现在的最优路径。

路径剪接。当一条路径需要被重新计算时，意味着世界正在改变。给定一个变化中的世界，地图上的邻近部分比远处的部分更好了解。我们可以遵循一个局部修正策略：找到附近的一条好路径，并且假定较远的路径直到我们靠近它了才需要重新计算

监视地图的改变。选择重新计算全部或部分路径在特定的时间间隔，是对地图的改变来触发重新计算。地图可以分成不同的区域，每个游戏单位可以在特定的区域表现出兴趣。

预测障碍物移动。如果障碍物的移动可以被预测，那就可以在进行寻路时把未来的障碍物位置纳入考虑。

5.NavMeshAgent组件的代理器移动到给定目标点需要利用哪个函数？

答：SetDestination(target)方法