达到最大内存限制时(maxmemory)，Redis 根据 maxmemory-policy 配置的策略，来决定具体的行为。

当前版本,Redis 3.0 支持的策略包括：6种

noeviction：不删除策略，达到最大内存限制时，如果需要更多内存，直接返回错误信息。大多数写命令都会导致占用更多的内存(有极少数会例外，如 DEL )。

allkeys-lru：所有key通用；优先删除最近最少使用(less recently used ,LRU) 的 key。

volatile-lru：只限于设置了 expire 的部分；优先删除最近最少使用(less recently used ,LRU) 的 key。

allkeys-random：所有key通用；随机删除一部分 key。

volatile-random：只限于设置了 expire 的部分；随机删除一部分 key。

volatile-ttl：只限于设置了 expire 的部分；优先删除剩余时间(time to live,TTL) 短的key。

如果没有设置 expire 的key，不满足先决条件(prerequisites)；那么 volatile-lru, volatile-random 和 volatile-ttl 策略的行为，和 noeviction(不删除) 基本上一致。

您需要根据系统的特征，来选择合适的驱逐策略。 当然，在运行过程中也可以通过命令动态设置驱逐策略，并通过 INFO 命令监控缓存的 miss 和 hit，来进行调优。

一般来说：

如果分为热数据与冷数据，推荐使用 allkeys-lru 策略。 也就是，其中一部分key经常被读写. 如果不确定具体的业务特征，那么 allkeys-lru 是一个很好的选择。

如果需要循环读写所有的key，或者各个key的访问频率差不多，可以使用 allkeys-random 策略，即读写所有元素的概率差不多。

假如要让 Redis 根据 TTL 来筛选需要删除的key，请使用 volatile-ttl 策略。volatile-lru 和 volatile-random 策略主要应用场景是：既有缓存,又有持久key的实例中。 一般来说，像这类场景，应该使用两个单独的 Redis 实例。

值得一提的是，设置 expire 会消耗额外的内存，所以使用 allkeys-lru 策略，可以更高效地利用内存，因为这样就可以不再设置过期时间了。