一、环形轨道，一辆车，车辆可以双向任意行驶。我们规定**车辆的原始位置为0**（该位置**也是车站1的位置**），按顺时针方向每个单位位置坐标加1。如果轨道总长为10，则按顺时针方向走，位置9的下一个为位置0。车站编号同理，也是按顺时针方向依次递增。**车速固定，每秒一个单位**。停车接人或乘客下车时需要**停车一秒钟**。无论一次停站完成几个服务停留时间统一为1秒钟。各站之间距离相等，车辆经过站点时，根据调度策略，车辆可以停也可以不停。其他位置不允许停车。车辆只能在站点停站时才能改变行驶方向。

二、各站之间距离可配置，站点个数可配置，调度策略可配置。这三个参数保存在配置文件中，程序要通过读配置文件获取。**配置文件的名字为dict.dic**。

配置文件为**文本文件**，以#号开头的行是注释，井号只可能出现在每一行的开头。

每行一个参数，格式为：

参数 = 值

的形式。每个参数前无空格，参数名、等号、参数值用空格分隔。

其中参数有三个，即TOTAL\_STATION，代表**站点总数，为大于1且小于等于20的整数**；DISTANCE，代表**每站之间的距离，为大于0且小于6的整数**；STRATEGY，代表调度策略，只能是FCFS（先来先服务），SSTF（最短寻找时间优先）和 SCAN（顺便服务）之一。

另外：

1、如果某个参数没有出现在配置文件中，则该参数取缺省值。

三个参数的缺省值如下：

TOTAL\_STATION = 5

STRATEGY = FCFS

DISTANCE = 2

2、**三个参数在文件中的顺序没有规定**。

3、显然，**TOTAL\_STATION与DISTANCE乘积就是轨道总长度**，所以配置文件中没有这个参数。

三、输入格式：

若干行，每行一个指令。

指令共5种。分别为end、clock、counterclockwise、clockwise 和target。

**其中end是程序退出指令（不是停运指令，是时钟停止，程序退出的意思），只在最后一行出现一次**；

clock是时钟指令，每出现一次代表过了一秒钟；

counterclockwise、clockwise、target为请求指令，如果它们出现，同一行内后边一定有一个整数。如果是counterclockwise和clockwise，代表站台上的请求，后边的整数代表请求发生的站点号，counterclockwise表示逆时针方向乘车请求，clockwise代表顺时针方向乘车请求。如果是target，代表车厢内下车请求，后边的整数代表要去的站点号。

四、输出格式：

程序开始，**先输出一次初始状态**，然后**每个clock输出一次当前状态**；程序**退出时输出end**。每次输出的格式如下，冒号后面没有空格，最后一次输出end后有一个换行。

**TIME:秒数**

**BUS:**

position:0

target:0000000000

STATION:

clockwise:0000000000

counterclockwise:0000000000

**首先输出当前的时间，即已过的秒数。**

然后三行代表车辆，BUS:固定不变，position:固定不变，后边的数字代表当前车辆位置，target:固定不变，后边一排数字依次代表车内站点请求情况，0表示没有请求，1表示有请求。

最后三行代表各站点的状态，STATION: 固定不变，clockwise: 固定不变，后边的数字依次代表各站点顺时针方向的请求情况，0表示没有请求，1表示有请求。counterclockwise: 固定不变，后边的数字依次代表各站点逆时针方向的请求情况，0表示没有请求，1表示有请求。

具体可参考输入样例。

五、策略补充说明：

1. 公交车在没有任何请求时，留在当前位置静止不动，处于空闲状态；只有收到上车或下车请求，确定服务目标后，才进入行驶状态。
2. 每一个请求均为单独的服务，即车内请求与站台请求没有必然联系。
3. 当公交车在空闲状态，或者完成上次请求 (到达目标站点并停满1秒) 后，开始一次新的调度。调度时，以之前收到的请求以及当前1秒内新发生的所有请求作为候选，按照调度规则选择出目标请求，整个调度过程执行时间忽略不计，即公交车在1秒的最开始完成上述所有动作，然后立即服务目标请求。
4. 当车服务目标时要选择路程短的方向行驶，如果两个方向路程相同则选择顺时针方向；
5. 如果在某个请求没有完成时再有相同的请求(请求类型和站点全部相同)发生，则该请求被抛弃。如果已完成的请求再次发生时应按新请求处理。
6. 对于先来先服务策略，车一次停站只完成一个请求，即使在这个站点上即有乘车请求，车内也有到该站的请求，也只能按策略完成已经调度的那个请求。但是完成当前请求后，如果发现时间序列上后续的一个或多个连续请求都恰好在同一站点（即连续的同站点请求位置相同，但请求类型不同），则可以立即完成这些连续的同站点请求，也就是说特殊情况下，一次停车的1秒内可完成多个请求。
7. 对于最短寻找时间优先策略，一次服务的目标请求一旦确定，即使中途产生更优的请求也不可以更改。但如果新的请求恰好可以顺便服务（同方向的站台请求或车内请求），可以为新的请求停站。具体为：程序计算离当前车的位置最近的请求，如果没有请求则原地不动，否则按最近的路线（顺、逆时针）去接（送）。如果车途中遇到与车目前同方向的上车或下车请求，可以停下一秒解决，反方向的上车请求不停车。车服务完目标后，反复此过程，直到end。特别地，当车到达目标站点时，可以停一次车(1秒钟)完成该站点已接收的所有类型请求(区别于顺便站停靠）。
8. 对于顺便服务策略。第一次行驶方向由时间最短请求站点决定。确定方向后，每次调度都按照当前方向，选择寻找时间最短的请求(不区分类型或方向)作为服务目标。如果去往服务目标站点行驶的距离超过轨道一半时，则需要切换行驶方向服务该请求。车辆行驶过程中，如果经过的站点有服务请求(上车或下车)，则不管这个请求的类型或方向，一律停站并完成此请求。这意味着一次停车可能同时完成3个服务请求(上车(顺时、逆时)和下车)。车辆没有请求时则原地不动，直到有新的请求时再按照上述规则继续运行。
9. 对于后两种策略(最短寻找时间、顺便服务)，如果车辆途经某站点本没有停车计划，则新的请求只要在车辆到达该站点前产生，就能允许停车服务。
10. 车处于停止状态开始一次新调度时（空闲状态或者完成上一次服务后），如果本站有请求且根据规则可以为该请求服务，则该请求立即完成，不再停1秒钟。

六、输入样例：（配置参数为TOTAL\_STATION = 10、STRATEGY = FCFS、DISTANCE = 3）

clock

counterclockwise 3

clock

clock

clock

clock

clock

clock

target 10

clock

clock

clock

clock

clock

clock

clock

clock

clock

clock

clock

end

七、输出样例：（配置参数为TOTAL\_STATION = 10、STRATEGY = FCFS、DISTANCE = 3）**TIME:0**

BUS:

position:0

target:0000000000

STATION:

clockwise:0000000000

counterclockwise:0000000000

**TIME:1**

BUS:

position:0

target:0000000000

STATION:

clockwise:0000000000

counterclockwise:0000000000

**TIME:2**

BUS:

position:1

target:0000000000

STATION:

clockwise:0000000000

counterclockwise:0010000000

**TIME:3**

BUS:

position:2

target:0000000000

STATION:

clockwise:0000000000

counterclockwise:0010000000

**TIME:4**

BUS:

position:3

target:0000000000

STATION:

clockwise:0000000000

counterclockwise:0010000000

**TIME:5**

BUS:

position:4

target:0000000000

STATION:

clockwise:0000000000

counterclockwise:0010000000

**TIME:6**

BUS:

position:5

target:0000000000

STATION:

clockwise:0000000000

counterclockwise:0010000000

**TIME:7**

BUS:

position:6

target:0000000000

STATION:

clockwise:0000000000

counterclockwise:0010000000

**TIME:8**

BUS:

position:6

target:0000000001

STATION:

clockwise:0000000000

counterclockwise:0000000000

**TIME:9**

BUS:

position:5

target:0000000001

STATION:

clockwise:0000000000

counterclockwise:0000000000

**TIME:10**

BUS:

position:4

target:0000000001

STATION:

clockwise:0000000000

counterclockwise:0000000000

**TIME:11**

BUS:

position:3

target:0000000001

STATION:

clockwise:0000000000

counterclockwise:0000000000

**TIME:12**

BUS:

position:2

target:0000000001

STATION:

clockwise:0000000000

counterclockwise:0000000000

**TIME:13**

BUS:

position:1

target:0000000001

STATION:

clockwise:0000000000

counterclockwise:0000000000

**TIME:14**

BUS:

position:0

target:0000000001

STATION:

clockwise:0000000000

counterclockwise:0000000000

**TIME:15**

BUS:

position:29

target:0000000001

STATION:

clockwise:0000000000

counterclockwise:0000000000

**TIME:16**

BUS:

position:28

target:0000000001

STATION:

clockwise:0000000000

counterclockwise:0000000000

**TIME:17**

BUS:

position:27

target:0000000001

STATION:

clockwise:0000000000

counterclockwise:0000000000

**TIME:18**

BUS:

position:27

target:0000000000

STATION:

clockwise:0000000000

counterclockwise:0000000000

end