```
     -2   -1  0   1   2    -> size=5
        |---|---|---|---|
```

position	step	final position	
2	1	-2	(2+1=3 is > size/2 ,so 3-5=-2)
2	2	-1	(2+2=4 is > size/2 , so 4-5=-1)
2	7	-1	(2+7=9 is > size/2 , so 9-5=4 ,4 is >size/2 , so 4-5=-1)
-1	1	0	(-1+1=0 is < size/2 , so leave it)
-1	3	2	(-1+3=2 is < size/2 , so leave it)

pos = -1; step = 4; fpos= -1 + 4 = 3; -(size/2)=-2.5; (size/2)=2.5; 3>2.5=>fpos=fpos-size=3-5=-2
pos= -1; step=-2; fpos=-1+-2=-3; -(size/2)=-2.5; -3<-2.5=>fpos=fpos+size=-3+5=2
if dsist >Length/2; dist=dist-length; if part intial pos. les than L/2 dist=-dist, if part intial
pos. greater than L/2, dist=positive if dist is new smaller dist (ie. dist >L/2) otherwise,
if dist<L/2 but part pos. from origin still greater than L/2 then dist is orginal dist. with
no necessary application of the periodic boundry conditions but is NEGATIVE.
How to account for the corners?? I think it will acont for the corners.
For sign perhaps you couls compare one particles position realativet ot the other,
almost like your not doing an abs(diff) dist.

LENGTH = 27.202644
WIDTH = 11.087434329

pos. part 1 = (x,z) = (3,5); pos. part 2 = (x,z) = (9,25)
applying perodicity
diff between part 1and part 2: x2-x1=9-3=6   z2-z1=25-5=20
diff between part 2 and part 1: x1-x2=3-9=-6   z1-z2=5-25=-20
For dist between part 1 and 2:
dist is in pos. vec. direction for z (left to right) and pos. for x (bot. to top)
apply pbc:
z>L/2? yes, so, z_dist= zdist-L, however opposite sign b/c z_dist was pos. vec. dirction
so z_dist now = 20-27=-7 if dist. vec was neg. it would be dist+L. d=-20,-20<-L/2
-basically checking each component and altering the distace vector.
for x: x_dist>W/2? yes, so, x_dsit now = 6-11=-5
so new dist is now (x,z)=(-5,-7) from what was (6,20)
I want to test to see if the image located (-5,-7) away is at the same postion relative
to the top right corners as the part 2 at (9,25):
so..11-9=2 and 27-25=2
part 1 at (3,5) so, -5+3=-2, -7+5=-2. So, they are at the same posit., respectively.
So, the necessary algorithim:
look at set of coordinates of parts. whose mag. dist. is to be calculated.
calc. diff. between each set of points depending on dist from 1->2 or 2->1 (also still calc. mag. dist)
I guess just gets the component form of the vector because some comps. will be negative
and others will not. So, take the pair of coords. stored in the tuple(for ex. to be used for
ist 1->2 and preform reverse operation for 2->1).
From there apply pbc by checking each component compared to >L/2. If e_dsit for comp. is pos.

then new dist for pbc will be e_dist-L bu if e_dist was neg. new dist.=e_dist+L.