**Bugs and Errors**

The **Bugs** we are talking about here are not the insects but the bugs in the computer program.

The **error** means there is some mistake in the code due which the code is not running at all. It can be due to multiple reasons like you forgot to define a variable, you forgot a bracket, you misspelled a function name etc. There are countless reasons due to which there can be errors in the code.

Now **bugs** are very interesting, if you have bugs in the code, your code may run, but it will have some different results than expected.

**Collision radius**

every sprite has a collision radius around it. It is like the skin of the sprite. It can experience touch. When two objects collide, their collision radius comes in contact. You can turn on the collision radius by using:

trex.debug = true;

We can make the collision radius small or large. A large collision radius will invoke the effect of collision even when there is no actual collision. The collision radius of our Trex is very large. We can set the collision radius of the Trex to what we desire.

**sprite.setCollider()** function is used to set the collider shape and size.

**sprite.setCollider(‘collider shape’, x offset, y offset, radius/width, height)**

Eg: trex.setCollider("circle",0,0,40);

trex.debug = true;

● The first argument is to set the shape of the collider.

● The second argument is for x-offset. x-offset is how far we want the center of the collider shape on x-axis from the center of the Trex animation.

● The third argument is for y-offset. y-offset is how far we want the centre of the collider in y-axis from the center of the Trex animation.

● The fourth argument is to set the radius of the Trex.

**Lifetime**

We assign lifetime to any game object by assigning it a specific number. After each frame, the lifetime reduces by 1. When lifetime becomes 0, the game object/sprite disappears.

New objects - clouds and obstacles - are continuously getting spawned. They continue to occupy space in the memory even when they are outside the screen. This causes a ‘**memory leak**’, if we don't destroy the objects we don't need to store.

in the END state, the obstacles don’t move - we want the obstacles to disappear after some time.

The number we should set as the lifetime of these game objects, so that they never disappear.

Remember how lifetime works. Every frame reduces the lifetime by 1 and the object disappears when lifetime is 0.

What number for lifetime will never reach 0 even when the lifetime is reduced/subtracted by 1 in every frame? A negative number

Exactly, if you set the lifetime of the game object to -1, with every frame, it will move away from 0 and never reach 0. We will need to assign the lifetime to a negative number to all the sprites in the ObstacleGroup and the CloudGroup when the gameState becomes END. We can set the lifetime of the sprites in the group using setLifetimeEach instruction at one go.

There is a small bug when we press space just at the time of collision

There is no gravity in the END state and we have just instructed the Trex to jump

We can set the trex y velocity to 0.