Week_15_CP

Trinity Introduction to Computer Programming With Game Development 01/10/2024

Todays class we will learn about Flow Charts, and why they are important to game developers, and have a critical think skill Hands on Exercise. Last we will briefly review JS Objects; then for PC Users - download Fenix, and the readme.txt if you have a MAC.

Then depending on time, we will learn about Numbering Systems and understand how:

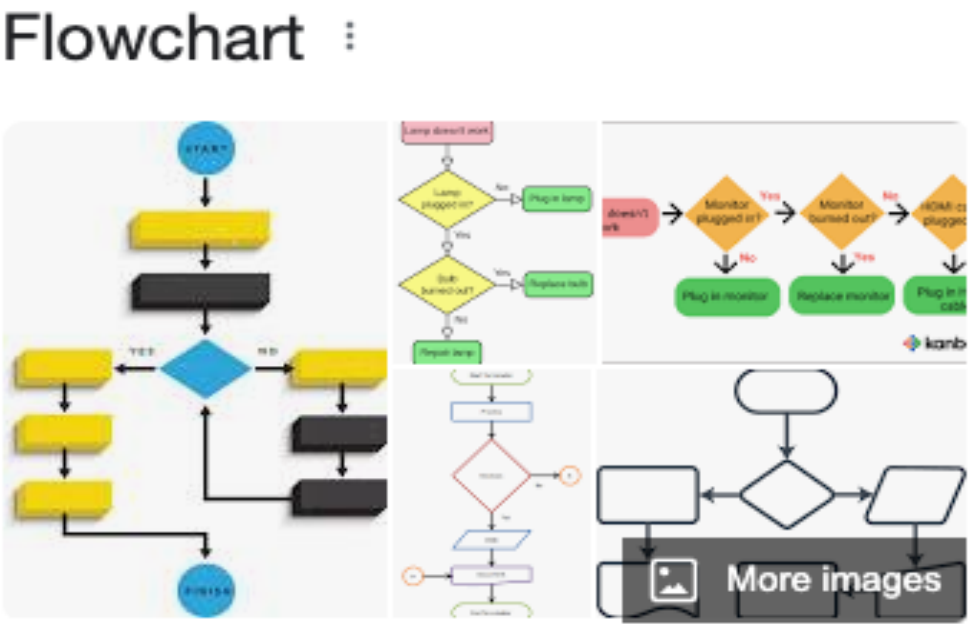
```
   10
 + 11
====
  101
```

(I know your thinking that ten plus eleven equals twenty one, how in the world can it be equal to 101?)

Today you will learn how this is true!

What is a Flowchart?

A Flowchart is a displayed representation of the steps to accomplish a specific task.



A flowchart is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task. The flowchart

shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. Wikipedia

4 Basic Flowchart Symbols for Creating a Flowchart

- The Oval. An End or Beginning While Creating a Flowchart. The oval, or terminator, is used to represent the start and end of a process. …
- The Rectangle. A Step in the Flowcharting Process. …
- The Arrow. Indicate Directional Flow. …
- The Diamond. Indicate a Decision.

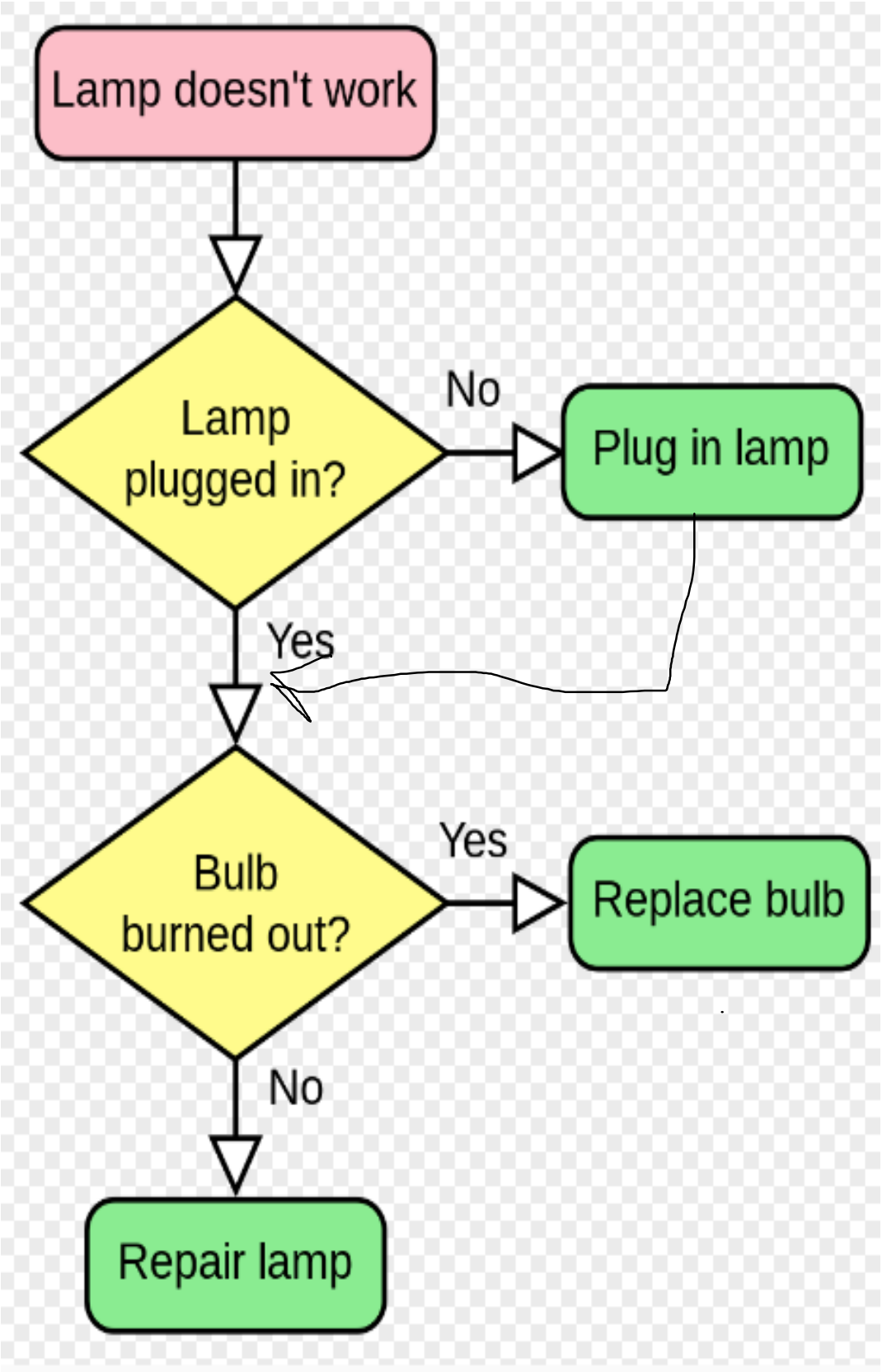
(Yes / No , True/False , 0 / 1)

Flowchart example:

Lets say your in your bedroom and your lamp stops working. Here is a flowchart to represent the steps you can take to fix the problem…

Remember Arrows indicate the flow.

Diamond is a decision, and decisions usually have a <Yes> and <No> flow to follow.

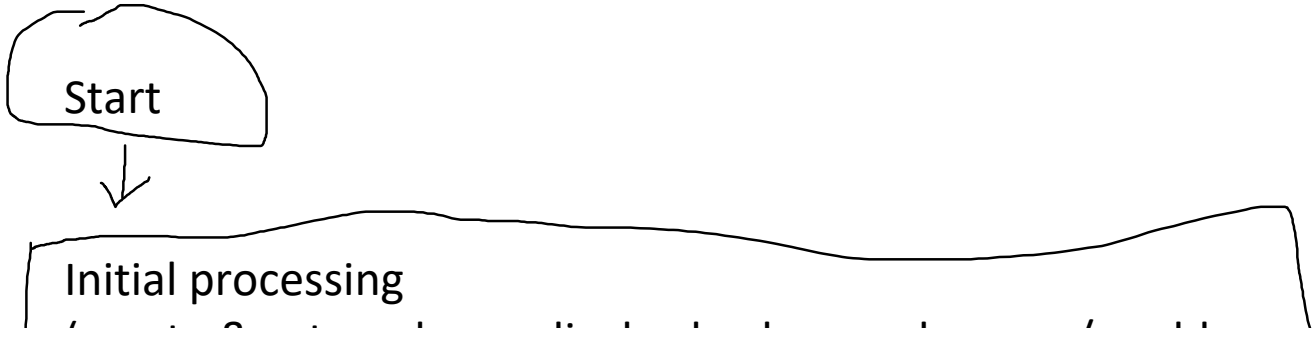

Lamp doesn't work
Lamp plugged in?
No
Plug in lamp
Yes
Bulb burned out?
Yes
Replace bulb
No
Repair lamp

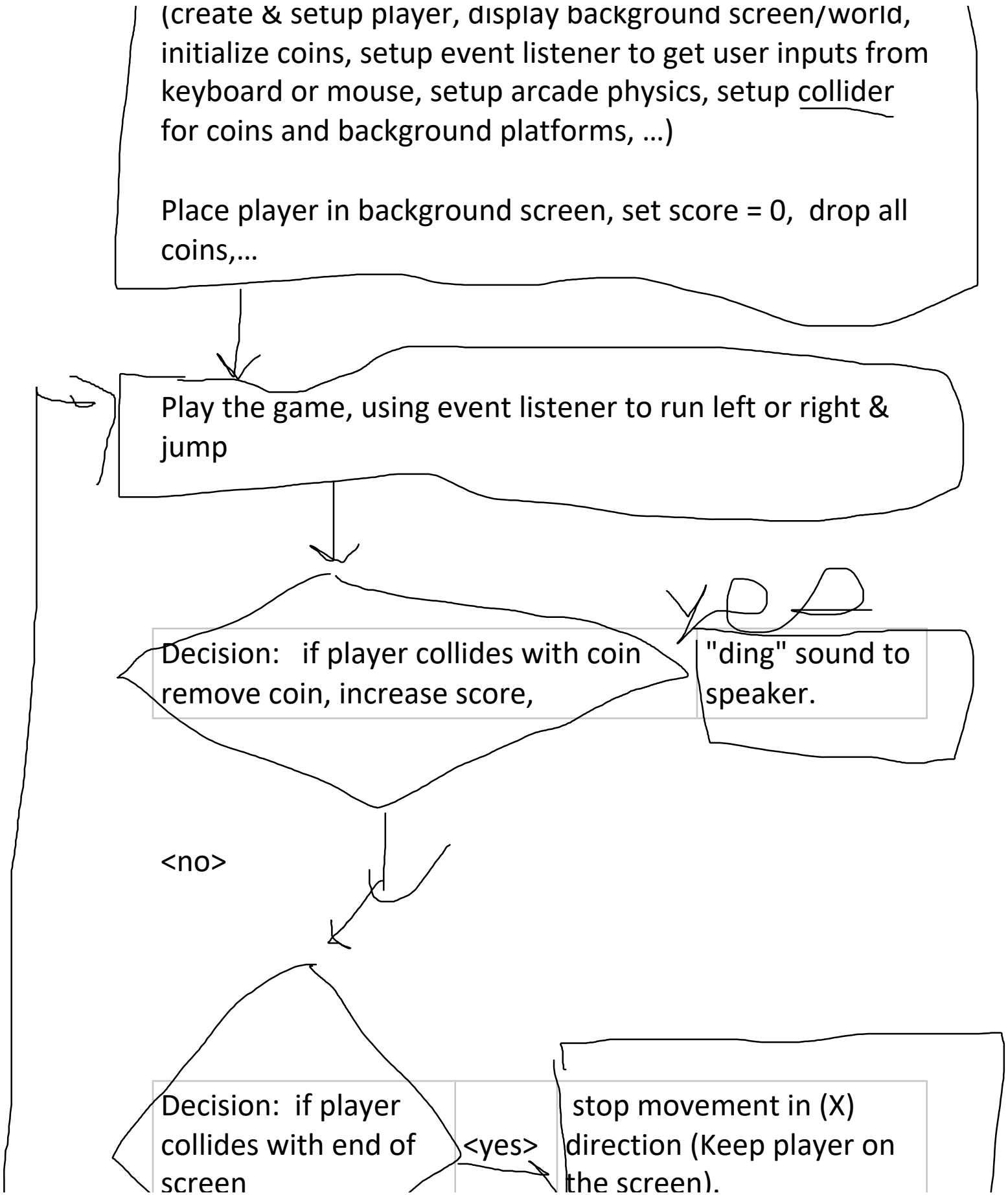
Notice: a flowchart is not linear

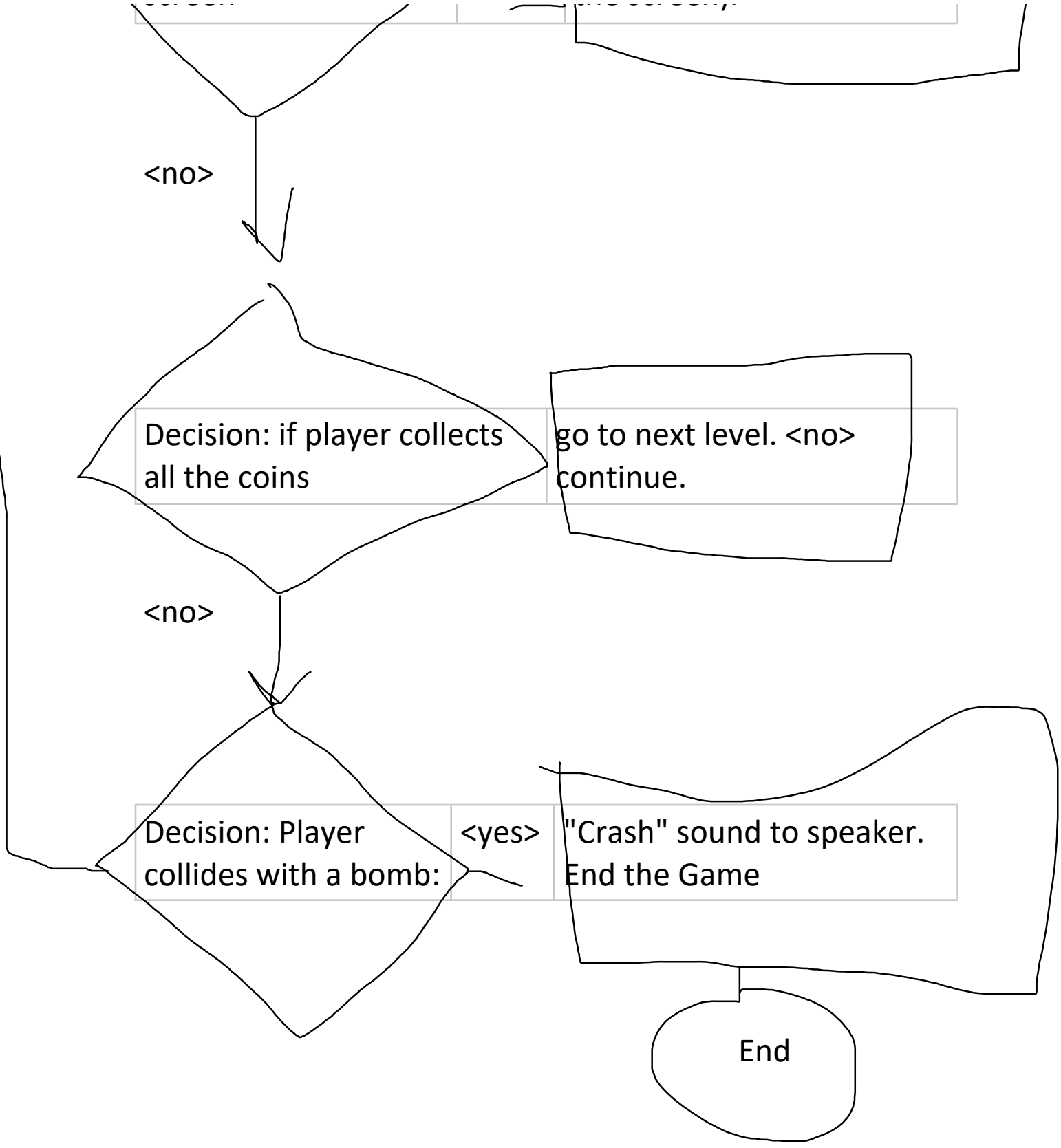
What steps would we take in this example:

We are designing a video game,
1 player that can run, jump, and collect coins,
2 in higher levels the player has to avoid bombs
that are floating by.

What would a flow chart look like for this:




(create & setup player, display background screen/world, initialize coins, setup event listener to get user inputs from keyboard or mouse, setup arcade physics, setup collider for coins and background platforms, …)
Place player in background screen, set score = 0, drop all coins,…
Play the game, using event listener to run left or right & jump
Decision: if player collides with coin remove coin, increase score,
"ding" sound to speaker.
<no>
Decision: if player collides with end of screen
<yes>
stop movement in (X) direction (Keep player on the screen).


<no>
Decision: if player collects all the coins
go to next level. <no> continue.
<no>
Decision: Player collides with a bomb:
<yes>
"Crash" sound to speaker. End the Game
End

Next we will have a short video:
https1://youtu.be/vBtGO9pXfrQ?si=v9Uq8JG6kQLb5RTq

JS is JavaScript, it is important for many reasons, one is that JS defines the behavior of Web Pages. Also it is the programming language we will use to develop our games.

Important to remember:

All Objects have properties & methods…

Object's Property:

Vehicle (Object)

Property : Number of wheels

4 = car

2 = motorcycle

3 = tricycle

How many seats?

What color?

How much gas is left?

Method: start the car

Move forward, turn left/right

Hit another object,

Go backwards…

*Methods are functions…

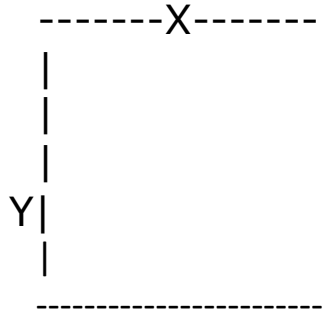
*Properties return values..

Canvas Coordinates

The HTML canvas is a two-dimensional grid.
The upper-left corner of the canvas has the coordinates (0,0)
In the previous chapter, you saw this method used: fillRect(0,0,150,75).
This means: Start at the upper-left corner (0,0) and draw a 150x75 pixels rectangle.

Coordinates Example

Mouse over the rectangle below to see its x and y coordinates:



JavaScript Classes

```
class Vehicle {
constructor(make, model, color) {
this.make = make;
this.model = model;
this.color = color;
}

getName() {
return this.make + " " + this.model;
 }
}
```

And in order to create a new instance of class `Vehicle`, we do this:

```
let car = new Vehicle("Toyota", "Corolla", "Black");
```

by writing the above code, we have actually created a variable named `Vehicle` which references to function constructor defined in the class, also we have added a method to the prototype of the variable `Vehicle`, same as bellow:

```
function Vehicle(make, model, color) {
    this.make = make;
    this.model = model;
```

```
    this.color = color;
}

Vehicle.prototype.getName()= function () {
    return this.make + " " + this.model;
}

let car = new Vehicle("Toyota", "Corolla", "Black");
```

So Class declarations are not `hoisted` means, you can't use a class before it is declared, it will return `not defined` error, see below:

This works:

```
> let car = Vehicle("Toyota", "Corolla", "Black");

  function Vehicle(make, model, color) {
      this.make = make;
      this.model = model;
      this.color = color;
  }
```

Homework:

Design a flowchart for a video game that you enjoy playing, include:

What needs to be loaded: (Players, world / display, score, achievements(positives), bombs or negative events, when game levels up, when game ends)…