Final Project: SaveThePresents
Due Thursday, December 4

You may work with a partner on this project.

Using `getElementById`, variables (count, etc.), functions, `setTimeout`, positioning, and parameters:

Note: The final project is a simple game in which you must get the hero to the present while avoiding the zombies (and other scary things). You must use arrow keys to move the hero around on the board. If you get the hero to the present, you get 10 points. If you get eaten by a zombie or run over by another bad thing, you turn into a splat and lose.

If you want to make a cooler game, feel free. It must use `getElementById`, variables, positions, `setTimeout`, and parameters. Have fun. Make the next angry birds or fruit ninjas (or, heck, minecraft!) Don’t let me hold you back.

HTML Part:
To create the Present game (or your version of it), create a web page with a Present image, a hero image, and a zombie image. You may wish to add another few scary images as well (e.g., I added a scary jack-in-the-box and a scary snowman image). Make sure all your images are the same width and height. The scary images will be moving across the screen automatically (you will write the functions for this in javascript later), so for now position them along the side of the web page (either side is fine). You will move the hero around the screen (using either arrow buttons or, if you prefer, keys on the keyboard). For now, assume you will be using the arrow keys, so in your html, below the images, create a table with 4 arrows – a left arrow, an up arrow, a right arrow, and a down arrow. Later in JavaScript you will write a function so that each of these arrows will move the hero image left 10 pixels, up 10 pixels, down 10 pixels, or right 10 pixels, respectively. These 4 arrows will call your javascript function with a parameter that will hold ‘left’, ‘up’, ‘down’, or ‘right’ depending on which arrow you pick.

Also include a paragraph that will hold your score and a button to start your game. Use CSS to style your page appropriately, with a background image and bold text.

So, as an example, my starting html looked like this:
**JavaScript Part:**

Now let’s start on the JavaScript.

The biggest challenge in this game is figuring out when your hero is over your present, and when it’s over the zombie. You do that by comparing the left position and the top position for the present and the hero, and also for the hero and the zombie. If both the left and top positions are within 15 pixels of each other (e.g.,

```javascript
if ((leftpospresent > (leftposhero – 15)) && (leftpospresent < (leftposhero + 15)) && (toppospresent > (topposhero – 15)) && (toppospresent < (topposhero + 15)))
```

the hero is at the present and should collect points. To clarify, && means “and”. The English version of the above javascript would be:

if the left position of the present is greater than the hero’s left position minus 15, and the left position of the present is less than the hero’s left position plus 15, and the top position of the present is greater than the hero’s top position minus 15, and the top position of the present is less than the hero’s top position plus 15 …

Inside the script (but outside the functions), you will need at least 7 variables (all initially set to 0):

- leftpospresent
- toppospresent
- leftposhero
- topposhero
- leftposzombie
- topposzombie
- totalscore

(6 of the above variables are for the positions of different images on your screen. If you add images, for each image you add, you will need to add two variables for each image: one for the position down from the top, and one for the position from the left.)

You want these variables outside the functions because you want them to be initialized once, and you want more than one function to be able to use these variables.

**Functions:**

1. **MovePresent function:**

Write a function that will move the present around the screen to random places. This is a short function.

- Set the leftpospresent to a random number between 0 and 800, and set toppospresent to a different random number between 0 and 800.
- Set the left and top positions, respectively, to the leftpospresent and toppospresent random variables.
- Then use setTimeout to recall the function every 20000 milliseconds (yes, that’s a big number. You want the present to stay still for a while so in the game you have time to navigate the hero to where the present is on the board. If you want to test this function, make it a smaller number. Then when you’re sure it works, set it back to that long number.)
- If you want, you can make the function move at a random number between every 10000 and 20000 milliseconds.

2. **Zombie function:**

Write a function that moves a zombie across the screen (like the trains did in class). You can assume your zombie only moves in one direction so this function will only change the leftposzombie. Since we’re assuming the screen
is about 800 pixels wide, you will want an if condition that only calls setTimeout if leftposzombie < 650 (assuming the zombie is about 150 pixels wide).

So if the leftposzombie is less than 650, you will want to increase the leftposzombie by some small amount (depending on how slow and how smooth you want the zombie’s movement to be), and then set the zombie image (in your html)’s left position to be leftposzombie. (Get this part of the function working first before you add the below code, because the code below won’t be testable until you add the MoveHero function).

Now here is where you want to check to see if the hero is at the same place as the zombie (in which case the zombie eats the hero and the hero loses all his points). So you must check to see if the hero is within 15 pixels of the zombie by checking to see if the leftposzombie and leftposhero are within 15 pixels of each other and the topposzombie and the topposhero are within 15 pixels of each other. You do this using this if condition:

```
if ((leftposzombie > (leftposhero – 15)) && (leftposzombie < (leftposhero + 15)) && (topposzombie > (topposhero – 15)) && (topposzombie < (topposhero + 15)))
```

(This should all be on one line, but MS Word won’t let that happen).

If the zombie and the hero are within 15 pixels of each other, the totalscore variable should get set to 0. Since this is the case for losing, you might also want to display a “game over” or “You’re a loser” image in place of the hero, but that’s your choice.

Otherwise (else) call setTimeout, with a relatively small time if the movement is small and a larger time if the movement is larger. You can play with these two values to get a very smooth-moving zombie, or a jerky-moving zombie. Your choice. Sometimes zombies look better when they move sort of jerkily.

Again, all this should only happen if the leftposzombie is < 650.

Otherwise (else) if leftposzombie >= 650, set leftposzombie to 0 and topposzombie to 0 and call the PlaceZombie() function.

**Move Other Scary Creature function:**

Write another function, almost exactly the same as the Zombie function, only this function will move one of your other scary creatures. In addition, this function will either move your other scary creature down the screen or from right to left (the opposite direction of the direction in which your zombie is moving). In this function you will also have to check whether this scary creature is over the hero or not and, if it is, you lose and totalscore is set to 0 (just like the Zombie function). Note that you can’t call this function Move Other Scary Creature – you will have to name it something more appropriate.

**Note Here:** You will need to write an almost identical function for every scary creature you have on your board. There is an easier way to do this with parameters. If you want to try it, go ahead. You will have to create parameters for the image’s id, for the leftpos of the scary creature, and for the top position of the scary creature. If you’d rather just create new functions, that is okay too.

**PlaceZombie function:**

The zombie function moves the zombie across the screen. But you probably want the zombie to move up and down the screen, so he’s not always going across in the same place. This function figures out how far down in the screen the zombie should be, and then calls the zombie function which will make the zombie move across the screen.

More specifically, this function should:
- Generate a random number between 0 and 650 (assuming the screen is 800 pixels wide and the zombie image is 150 pixels high).
- Set topposzombie to be the random number.
- Change the zombie image on the web page’s src to be the zombie.gif (or whatever you called it).
- Change the zombie image’s position from the top to topposzombie.

Now you have a zombie that is a certain distance down from the top. Finally, in this function you should:

- Call the Zombie() function (this will move the zombie across the screen at its new position down from the top)

Note that this function doesn’t use a setTimeout to call itself because the Zombie function calls it instead.

**Place Other Scary Creature function:**
Write another function, almost exactly the same as the PlaceZombie function, only this function will place one of your other scary creatures by generating a random number and either placing its left position (if your scary creature is descending from the top) or placing its top position (if your scary creature is moving from right to left).
This function will need to call your Other Scary Creature function, just like the PlaceZombie function needed to call the Zombie function. Note that you can’t call it Place Other Scary Creature – you will have to name it something more appropriate.

**Note Here:** You will need to write an almost identical function for every scary creature you wrote a function for. There is an easier way to do this with parameters. If you want to try it, go ahead. Again, you will need to use parameters so JavaScript knows what scary creature you’re moving.

**MoveHero Function:**
- Write a function that will move the hero to the left 10 pixels (you just need to subtract 10 from the leftposhero.) Make your left arrow image call this function. Make sure it works.
- Now modify this function by adding a parameter. Modify your call to the function (in the left arrow) so that the call holds the word ‘left’. And add a condition in the function saying, if the parameter holds the word ‘left’, then subtract 10 from leftposhero (this is a simple if condition.) Make sure this works.
- Once you have this, modify it so that the right arrow image calls the same function, only with the word ‘right’. Inside the function, add another condition: if the parameter holds the word ‘right’, add 10 to leftposhero. Make sure this works so that the right arrow moves the hero right when you click on it.
- Do the same for the up and down arrows – if you click on the up arrow, have the hero move up 10 pixels. If you click on the down arrow, have the hero move down 10 pixels.
- Make sure all your arrows work to move the hero around. Once you have all your arrows working, you’re in great shape. Most of the program is done.
- Now check to see if the hero is at the same position as the present. You do that by looking to see if the leftposhero and leftpospresent and the topposhero and the toppospresent are within 15 pixels of each other. (almost exactly like you did in the zombie function, e.g.:
- if ((leftposhero > (leftpospresent – 15 ))&& (leftposhero < (leftpospresent + 15))...)  
If they are within 15 points of each other you want to increase the totalscore by 10, and use getElementById to write to the paragraph that will hold the total score the updated score.

**StartIt() function:**
This is the final function. The only reason you need this function is because when you call a function by clicking on a button, only one function can be started. But when you click on the start button in your html code, you want the MovePresent() function to start and you want the PlaceZombie function to start so that both the present is placed on the screen and the zombie starts moving across the screen simultaneously. If you have other creatures moving
across the screen, you will want those functions to start simultaneously (with the MovePresent and PlaceZombie functions) as well.

Button – only calls one function. Game – needs more than one function to start. Dilemma.

Solution: have the button call one StartIt() function, and have the StartIt function call all the functions you want to start simultaneously. Without the other scary creature functions, your StartIt function should consist of 2 lines and look like this:

```javascript
function StartIt()
{
    MovePresent()
    PlaceZombie()
}
```

You will need to add at least one more line for the Place Other Scary Creature function you wrote (depending on what you call it).

This is the function that should be called when the start button in the web page is clicked on.

That’s it! You’ve written the SaveThePresents game!

A quick note: Every programming language has variables, arrays, if conditions with else if and else options, functions, parameters, and loops (setTimeout is a way of looping, or having something happen again and again). You can apply the skills you have used on this project to any programming activity you elect to take on in the future.

Extra Credit a: (5 pts):

Use keyboard input instead of clicking on arrows to move the hero around the board (see the optional tutorial). This makes playing the game a lot easier.

Extra Credit b (5 – 30 pts, depending on level of difficulty and effort):

There are a number of ways in which you could make this game more polished, exciting, and interesting. For instance, you could make the monsters (zombies, etc) move across the screen at random speeds, so their movements would be less predictable. You could add more monsters, possibly moving diagonally. You could choose to have a limited number of presents, and if the hero is able to collect that many presents, s/he wins the game. You could make the present fade when the hero collects it, and have a paragraph on top that shows how many presents have been collected and how many still need to be collected. You could make the hero’s image change to a chomped image when the zombie eats the hero, and/or make the hero’s image change to fireworks or something like that when s/he successfully get the present. These are just ideas I thought of quickly, and I’m a lousy gamer, so I’m sure you can come up with some better ideas.

Extra Credit will be given for taking this game to the next level. If you can think of cool upgrades to the game, let me know. They may be worth extra credit!