

Computer Science

Year 12 into year 13 summer independent work

Deadline – first lesson in September

Part 1 – Website Development

HTML, CSS, JavaScript Programming Task

As there will be a section of coding on your exam that relates to HTML, CSS and JavaScript an exercise has been devised that will require you to develop a website that uses all three languages.

Theme

Produce a website on a topic of your choice. This could be your favourite film, your favourite brand, a computer game or any other suitable topic. Your website will need to have a minimum of four HTML pages in it.

Requirements

1. Your website

Language Features to be Used

HTML

```
<!-- awareness of the following tags. Any other tags used will be introduced in the question.

<html>
<link> to link to a CSS file
<head>

<title>

<body>

<h1> <h2> <h3>

<img> including the src, alt, height and width attributes.

<a> including the href attribute.

<div>

<form>
<input> where the input is a textbox (i.e. has the attribute type="text" and another attribute name to identify it) or a submit button (i.e. has the attribute type="submit")

<p>
<li>
<ol>
<ul>
<script>
```

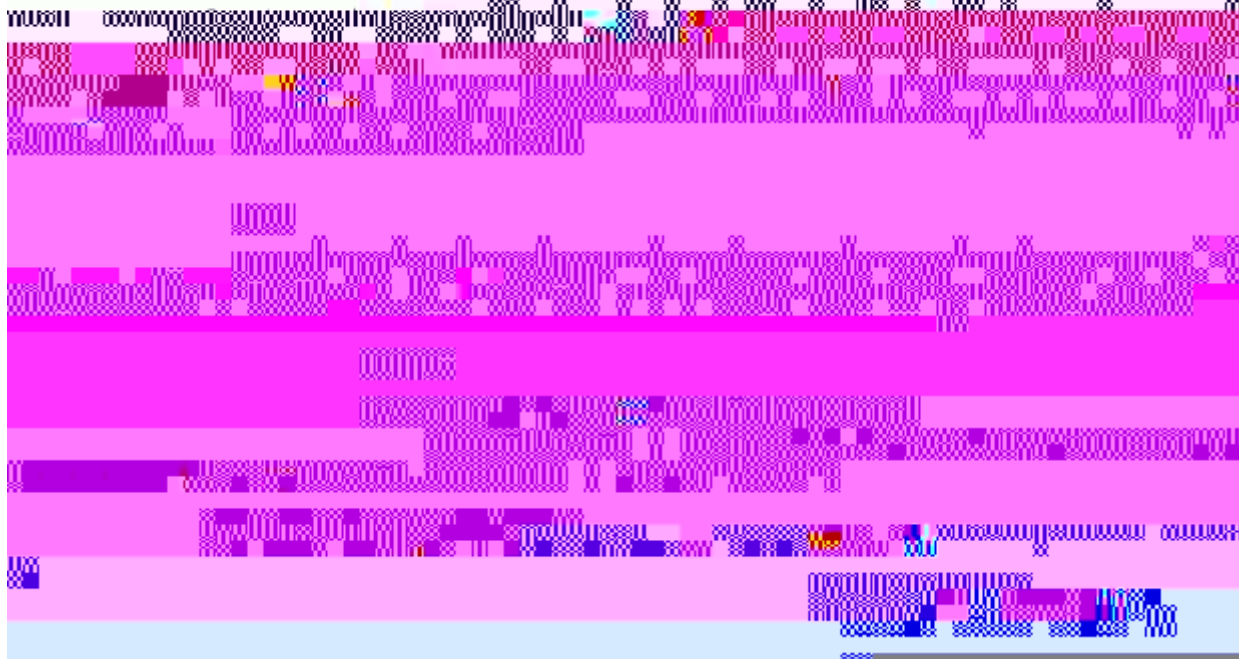
CSS

```


```

JavaScript

Learners are expected to be able to follow a list of tasks that will get practical experience of JavaScript in the following areas:



// Help is easily available via the internet, but I recommend w3schools if you do need help.

<https://www.w3schools.com/html/default.asp>

<https://www.w3schools.com/css/default.asp>

<https://www.w3schools.com/js/default.asp>

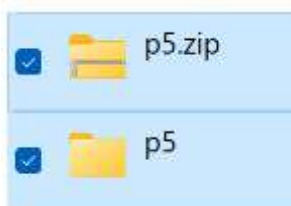
Part 2 – JavaScript

As shown on the previous page, the exam board information about JS is quite vague, so we are going to complete a number of exercises to give you experience of programming a variety of familiar algorithms using the JavaScript syntax.

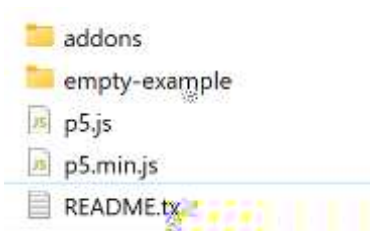
Exercise 1 – Visit <https://p5js.org/download/> and then click on the button highlighted pink below.



You should get a zip file that you can extract/unzip as shown below:

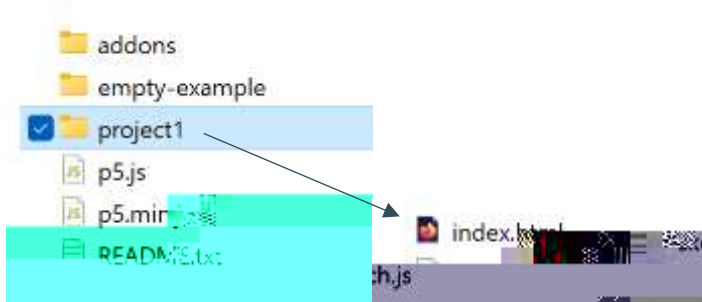


2 – Put the two files into your chosen folder and then navigate to that location, inside the p5 folder you should see:



This is the p5 library plus in the addons folder there is additional functionality for sounds – we won't need this however.

The empty-example folder is the base folder and contains a .html file and a .js file. I would make a copy and rename it (for example) project1, and then every time you want to make a new project, you can copy and paste the empty-example folder to start your next mini project.



Using an IDE (I am using sublime text - link in part 1 above) go to file > open and open both of the index.html and also sketch.js. They should look like this:

Index.html:

```
index.html sketch.js
1 <!DOCTYPE html>
2 <html lang="">
3
4 <head>
5   <meta charset="utf-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7
8   <script src="p5.js"></script>
9   <script src="p5.min.js"></script>
10
11 <body>
12   <div style="background-color: #f00; width: 100px; height: 100px; margin: 0 auto; border: 1px solid #f00; border-radius: 50%;>
13     <div style="background-color: #00f; width: 50px; height: 50px; margin: 0 auto; border: 1px solid #00f; border-radius: 50%;>
14
15   </div>
16 </body>
17 </html>
```

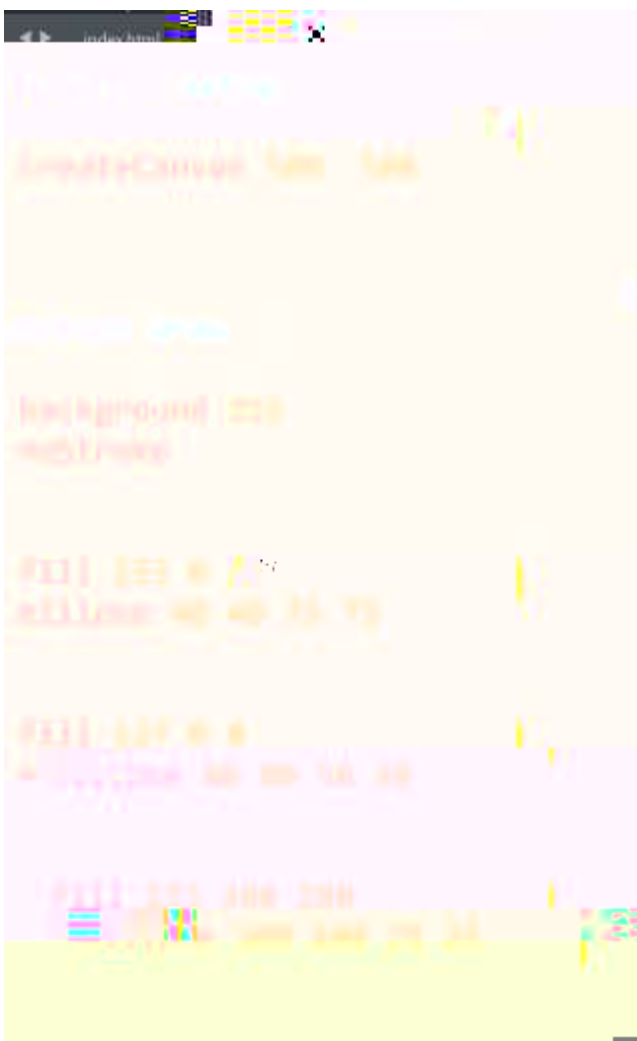
You don't need to do anything with this index file, it already points to the p5 library and the sketch file where you will add your code.


Sketch.js:

```
1 function setup() {  
2   // put setup code here  
3 }  
4  
5 function draw() {  
6   // put drawing code here  
7 }
```

This is the file that we will edit to learn some simple js by creating code to draw and then display in our browser.

Type the following code, save it, and then test it by opening the index.html in a browser - I am using FireFox.





This is what you should see in a browser. Paste your code/proof onto your evidence document.

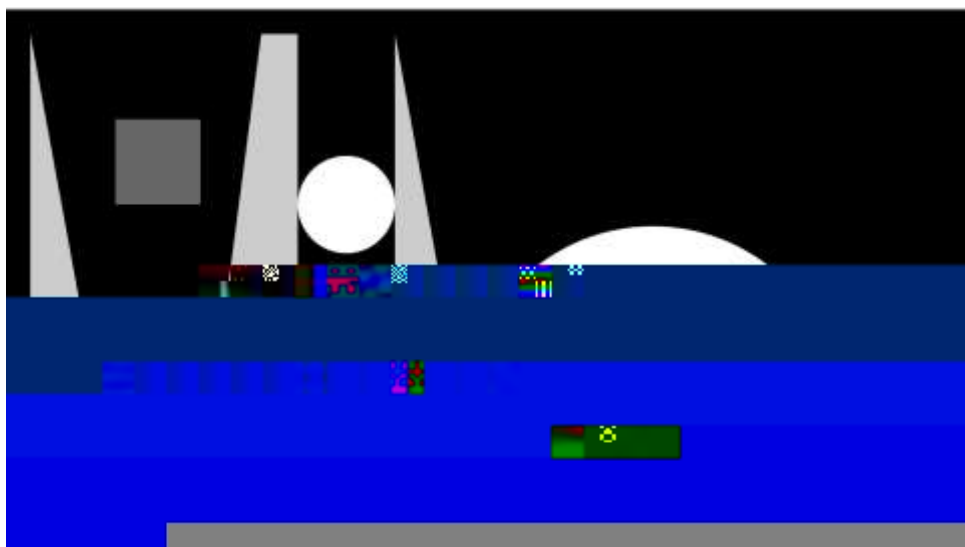
Q - What do the parameters do that we pass to the ellipse function? Show a variety of other ellipses.

Exercise 2 - Start a new project by copy and pasting empty-

Exercise 4 - Code the shapes below.

```
1 function setup() {
2   // Sets the screen to be 720 pixels wide and 400 pixels high
3   createCanvas(720, 400);
4   background(0);
5   noStroke();
6
7   fill(204);
8   triangle(18, 18, 18, 360, 81, 360);
9
10  fill(102);
11  rect(81, 81, 63, 63);
12
13  fill(204);
14  quad(189, 18, 216, 18, 216, 360, 144, 360);
15
16  fill(255);
17  ellipse(252, 144, 72, 72);
18
19  fill(204);
20  tria
```

Results in:



Pay special attention to the `arc()` function, the function draws part of an ellipse, the parameters work similar to other shapes, but have start and stop points measured in radians.

- Parameter 1 - x location
- Parameter 2 - y location
- Parameter 3 - width
- Parameter 4 - height
- Parameter 5 - start (in radians)
- Parameter 6 - stop (in radians)

- QUARTER_PI = 45 degrees
- HALF_PI = 90 degrees
- PI = 180 degrees
- PI + HALF_PI = 270 degrees
- TWO_PI = 360 degrees

```
1 function setup() {  
2   // Sets the screen to be 720 pixels wide and 400  
   createCanvas(720, 400);  
   background(0);  
   noStroke();  
   fill(255);  
   arc(200, 300, 200, 200, quarter_PI, TWO_PI - quarter_PI);  
}
```

Results in:



Exercise 5 - Create a house scene using shapes, and make it multicoloured rather than monotone as in the example above.

Exercise 6 - Combine the code we have done so far to make a moving PacMan style shape that appears animated and closes its mouth on each movement.



The code below is code for an insertion sort, and is written in python.



Exercise 8 - Use the code from the two examples to create a visualisation of the insertion sort in p5.js.

Exercise 9 - Quick sort - use the link or code from scratch using the code below:

<https://p5js.org/examples/simulate-quicksort.html>

```
// width of each bar is taken as 8.
let values = [];
// The array 'states' helps in identifying the pivot index
// at every step, and also the subarray which is being sorted
// at any given time.
let states = [];

// The setup() function is called once when the program
// starts. Here, we fill the array 'values' with random values
// and the array 'states' with a value of -1 for each position.
function setup() {
  createCanvas(710, 400);
  for(let i = 0; i < width/8; i++) {
    values.push(random(height));
    states.push(-1);
  }
}
```

```
    quickSort(0, values.length - 1);  
}
```


```
// The statements in draw() function are executed continuously  
// until the program is stopped. Each statement is executed  
// sequentially and after the last line is read, the first  
// line is executed again.
```

```
function draw() {
```

```
// restore original state
states[index] = -1;
await Promise.all(
  [quickSort(start, index - 1),
   quickSort(index + 1, end)
  ])states
}

// We have chosen the element at the last index as
// the pivot element, but we could've made different
// choices, e.g. take the first element as pivot.
async function partition(start, end) {
  for (let i = start; i < end; i++) {
    // identify the elements being considered currently
    states[i] =
```





Exercise 10 – Merge sort

The code

