1. What is **SASS**? What does **SASS** stand for?
2. What is a **CSS pre-processor**?

A **CSS preprocessor** is a program that lets you generate CSS from the preprocessor's own unique syntax. There are many CSS preprocessors to choose from, however most CSS preprocessors will add some features that don't exist in pure CSS, such as mixin, nesting selector, inheritance selector, and so on. These features make the CSS structure more readable and easier to maintain.

Few of most popular CSS preprocessors

[Sass](https://sass-lang.com/)

[LESS](https://lesscss.org/)

[Stylus](https://stylus-lang.com/)

[PostCSS](https://postcss.org/)

1. What does a **pre-processor** have to do with **SASS**?
2. Why use **SASS**?

**Sass** enables you **to use** things like variables, nested rules, inline imports and more. It also helps to keep things organised and allows you to create style sheets faster. **Sass** is compatible with all versions of CSS.

1. **SASS** has disadvantages? Which are?
2. What is a **SASS Variable**? Explain why are useful

A variable allows you to store a value or a set of values, and to reuse these variables throughout your SASS files as many times you want and wherever you want. Easy, powerful, and useful.

$blue: #004BB4;

$ubuntu-font: 'Ubuntu', 'Arial', 'Helvetica', sans-serif;

h1 {

font: $ubuntu-font;

color: $blue;

}

1. Explain the SASS **variables** property with an example.
2. What is a **mixin**? Why is it important? Give an example

Mixins are like functions in other programming languages. They return a value or set of values and can take parameters including default values. Note that SASS also has functions, so do not confuse a mixin with a function.

@mixin set-font( $family: 'Ubuntu' , $weight: 400 , $style: normal ) {

font-family: $family , 'Arial', 'Helvetica', sans-serif;

font-style: $style;

font-weight: $weight;

}

h1 {

@include set-font;

color: $blue;

}

If you want to update the default mixin values, you just need to pass the parameters within the @include call.

h1.callout {

@include set-font('Nunito',600);

color: $blue;

}

1. What is **SCSS**? Give an example
2. What is **SASS**? Give an example

SASS (Syntactically Awesome Stylesheets) is a CSS pre-processor that lets you use variables, mathematical operations, mixins, loops, functions, imports, and other interesting functionalities that make writing CSS much more powerful. In some ways, you may think of SASS as a style sheet extension language because it extends the standard CSS characteristics by introducing the benefits of a basic programming language. So SASS will compile your code and generate the CSS output a browser can understand.

$font-stack: Helvetica**,** sans-serif;

$primary-color: #333;

body {

**font**: 100% $font-stack;

**color**: $primary-color;

}

1. What is the difference between **.scss** and **.sass** syntax.
2. In which cases would we use **SCSS**? And in which cases would we use **SASS**?

**SASS** is used when we need an original syntax, code syntax is not required for **SCSS**. **SASS** follows strict indentation, **SCSS** has no strict indentation. **SASS** has a loose syntax with white space and no semicolons, the **SCSS** resembles more to **CSS** style and **use** of semicolons and braces are mandatory.

1. Explain how **traditional CSS** and **Preprocessed CSS** **workflows** are different.
2. Can we create functions with **SASS**? If it is true, give an example.

Functions allow you to define complex operations on [SassScript values](https://sass-lang.com/documentation/values) that you can re-use throughout your stylesheet. They make it easy to abstract out common formulas and behaviors in a readable way.

Functions are defined using the @function at-rule, which is written @function <name>(<arguments...>) { ... }. A function’s name can be any Sass identifier. It can only contain [universal statements](https://sass-lang.com/documentation/syntax/structure#universal-statements), as well as the [@return at-rule](https://sass-lang.com/documentation/at-rules/function#return) which indicates the value to use as the result of the function call. Functions are called using the normal CSS function syntax.

**@function** **pow**($base**,** $exponent)

$result: 1

**@for** $\_ **from** 1 **through** $exponent

$result: $result **\*** $base

**@return** $result

**.sidebar**

**float**: left

**margin-left**: **pow**(4**,** 3) **\*** 1px

1. What is **nesting**? Is it useful? Give an example of **nesting**
2. Difference between **@use & @import**? Give an example

The new @**use** is similar to @**import** . but has some notable differences: The file is only **imported** once, no matter how many times you @**use** it in a project. ... Similarly, @extends will only apply up the chain; extending selectors in **imported** files, but not extending files that **import** this one.

1. How can we **import other CSS/SASS files** in SASS? Give an example
2. Explain the concept of **inheritance** in **SASS.**

**Using @extend lets you share a set of CSS properties from one selector to another. It helps keep your Sass very DRY. In our example we're going to create a simple series of messaging for errors, warnings and successes using another feature which goes hand in hand with extend, placeholder classes. A placeholder class is a special type of class that only prints when it is extended, and can help keep your compiled CSS neat and clean.**

1. Why use **@extend?** Give an example