**WorkAround**

WorkAround is a human resources program that returns basic information about an employee, based on salary data. Using WorkAround, one can use an employee’s salary data to:

* return the cadre of the of the employee, in other words, whether this employee is entry level, mid level, or senior level.
* calculate employee tax rates
* return employee benefits
* calculate employee bonuses
* calculate the total amount an employee can be reimbursed based on the total value of their health, housing, or wellness benefits

WorkAround currently contains the data and functions in a single file, though they would like to modify the program so it makes use of JavaScript modules. Specifically, WorkAround asks you to demonstrate the different ways to export and import modules.

If you get stuck during this project or would like to see an experienced developer work through it, click “**Get Help**“ to see a **project walkthrough video**.

**Tasks**

**1/17Complete**

Mark the tasks as complete by checking them off

**1.**

Run the program in **employee.js** to view the output in the console.

Notice that the getEmployeeInformation() function at the bottom of the file returns the cadre, tax, benefits, bonus, and reimbursement eligibility based on a person’s payGrade.

Hint

Press the Run button to run the program.

**2.**

Let’s transform the program using modules.

We’ll first segment the program by placing the behavior and data in **employee.js**, and the use of the data in **workAround.js**. Copy the getEmployeeInformation() function and the three function calls beneath it, and paste the function into the **workAround.js** file. Be sure to remove the copied code from **employee.js** completely.

Notice that when you run the program, the console returns a ReferenceError that salary is not defined. We’ll fix this as we build out the program with modules.

Hint

Remove this code from **employee.js** and place it in **workAround.js**.

function getEmployeeInformation(inputSalary) {

...

}

getEmployeeInformation(...);

getEmployeeInformation(...);

getEmployeeInformation(...);

**3.**

Let’s turn the contents of the information in **employee.js** into a module. At the top of the file, using let, set the Employee to an object that represents the module.

You will not see any output yet.

Hint

let Employee = {};

**4.**

Move the salary property inside the Employee object, keeping the default value of 100000, using the appropriate syntax for defining a property and its value.

Recall that we can also set properties inside the object, and it has the same value as if it was declared outside the object as Employee.salary.

We’ll be declaring salary inside the Employee object in this project.

Hint

Employee = {

salary: 100000

}

**5.**

For the rest of the functions in **employee.js**, define getCadre(), calculateTax(), getBenefits(), calculateBonus(), and reimbursementEligibility() as properties on the Employee object. If you are uncertain about the proper syntax for this, check the hint.

We do not need to define payGrades as a property of the employee module for this project.

You will not see output yet.

Hint

The functions should be written in this format:

Employee.calculateTax = function() {

return payGrades[Employee.getCadre()].taxMultiplier \* Employee.salary;

}

...

**6.**

In **employee.js**, within the body of each function, closely check that each property of the Employee object is defined on Employee. For example, every instance of salary should be defined as Employee.salary since it is a property of the Employee object.

Note that you will not see output yet.

Hint

Be sure to change six instances of salary to Employee.salary and three instances of getCadre() to Employee.getCadre().

However, no instances of payGrades will change and the variable will be used as declared.

**7.**

We’ll now export the module. In **employee.js**, at the very bottom of this file, export the module Employee module using the export default syntax.

Hint

export default Employee;

**8.**

In **workAround.js**, import the Employee module using the import statement. Recall that you will need to import the module using the ./employee file path.

Note that you will still see the ReferenceError at this point.

Hint

import Employee from './employee';

**9.**

In **workAround.js** in the getEmployeeInformation() function, modify the variables such that they are a property on the Employee module.

For example salary should be defined as Employee.salary and calculateTax() should be modified to Employee.calculateTax().

You should see the program’s output.

Hint

function getEmployeeInformation(inputSalary) {

Employee.salary = inputSalary;

console.log('Cadre: ' + Employee.getCadre());

console.log('Tax: ' + Employee.calculateTax());

console.log('Benefits: ' + Employee.getBenefits());

console.log('Bonus: ' + Employee.calculateBonus());

console.log('Reimbursement Eligibility: ' + Employee.reimbursementEligibility() + '\n');

}

**10.**

Nice job! WorkAround now asks you to demonstrate how to export both traditional variables and variables as aliases.

In order to do this, we’ll need to modify each of the functions, so they are no longer properties on the Employee object.

You will not see output at this point.

Hint

Modify the functions so that they follow this format:

function getCadre() {

...

}

**11.**

You’ll also need to make sure that each mention of a function is no longer a property on the Employee object.

Specifically, getCadre() will no longer be a property of the Employee object.

However, you can leave salary as Employee.salary since it is inside the object.

**12.**

Now, in **employee.js**, modify the export statement such that you export the variables as the following:

* Employee with no alias
* getCadre an alias of cadre
* calculateTax an alias of tax
* getBenefits an alias of benefits
* calculateBonus an alias of bonus
* reimbursementEligibility an alias of reimbursement

Hint

export { Employee, getCadre as cadre, calculateTax as tax, getBenefits as benefits, calculateBonus as bonus, reimbursementEligibility as reimbursement };

**13.**

In **workAround.js**, import the following variables.

* Employee
* cadre
* tax
* benefits
* bonus
* reimbursement

When you run the code, you will get a TypeError.

Hint

import {Employee, cadre, tax, benefits, bonus, reimbursement} from './employee';

**14.**

In **workAround.js** in the, getEmployeeInformation() function, modify the function, such that each of the variables is written as its alias. For example, Employee.getCadre() will be modified to cadre().

You should see the program’s output!

Hint

function getEmployeeInformation(inputSalary) {

Employee.salary = inputSalary;

console.log('Cadre: ' + cadre());

console.log('Tax: ' + tax());

console.log('Benefits: ' + benefits());

console.log('Bonus: ' + bonus());

console.log('Reimbursement Eligibility: ' + reimbursement() + '\n');

}

**15.**

Nice work! WorkAround asks you demonstrate one more thing: combine both exporting variables as they are declared and the export default statement.

In **employee.js** use the export keyword to export getCadre(), calculateTax(), getBenefits(), calculateBonus() and reimbursementEligibility() as they are declared.

At the bottom of the file, use the export default statement to export the Employee module.

Hint

export let payGrades = {

...

};

export function getCadre() {

...

}

export function calculateTax() {

...

}

export function getBenefits() {

...

}

export function calculateBonus() {

...

}

export function reimbursementEligibility() {

...

};

export default Employee;

**16.**

In **workAround.js**:

* Write an import statement that imports each of the variables using their variable names.
* Beneath this statement, write another import statement that imports the Employee module.

Your code will return a ReferenceError. We’ll resolve this in the next step.

Hint

import { getCadre, calculateTax, getBenefits, calculateBonus, reimbursementEligibility } from './employee';

import Employee from './employee';

**17.**

In a last and final step, in **workAround.js**, modify the getEmployeeInformation() function, such that it refers to the varables that were exported as declared as well as the Employee module.

You should see the program’s output!

Hint

function getEmployeeInformation(salary) {

Employee.salary = salary;

console.log('Cadre: ' + getCadre());

console.log('Tax: ' + calculateTax());

console.log('Benefits: ' + getBenefits());

console.log('Bonus: ' + calculateBonus());

console.log('Reimbursement Eligibility: ' + reimbursementEligibility() + '\n');

}