Function in Javascript Assignment Answer

Q1. Create an arrow function called square that takes a number as an argument and returns its square. Use the arrow function to calculate the square of a given number and display the result.

Ans:-
```
const square = (value) => {
   return value ** 2;
}
console.log(square(5));
```

Q2. Create a JavaScript function called generateGreeting that takes a name as an argument and returns a personalized greeting message. Use this function to greet three different people.

Ans:-
```
function generateGreeting(name = "Dear"){
   console.log(`\nHello ${name}, Welcome to PW skils.`);
}
generateGreeting();


generateGreeting("Ashish");
generateGreeting("Vinay");
generateGreeting("Shanket");
```

Q3. Create an IIFE (Immediately Invoked Function Expression) that calculates the square of a number and immediately displays the result.

Ans:-
```
(function(){
   function square(value){
      console.log(value**2);
```

```
  }
  square(5);
})()
```

Q4. Write a JavaScript function called calculateTax that takes an income as an argument and returns the amount of tax to be paid. Use a closure to handle different tax rates based on income ranges. Test the

function with various incomes.

Ans:- function calculateTax() {

```
  // Define tax rates and income ranges using closures
  const taxRates = [
    { range: [0, 250000], rate: 0 },
    { range: [250001, 500000], rate: 0.05 },
    { range: [500001, 1000000], rate: 0.1 },
    { range: [1000001, 5000000], rate: 0.2 },
    { range: [5000001, Infinity], rate: 0.3 }
  ];

  // Return a function that calculates tax based on income
  return function(income) {
    // Find the appropriate tax rate based on income range
    const applicableRate = taxRates.find(({ range }) => income >=
range[0] && income <= range[1]);

    if (applicableRate) {
      // Calculate and return the tax amount in Rupees
      return income * applicableRate.rate;
    } else {
      // Handle invalid income ranges
      throw new Error('Invalid income range');
    }
  };
}

// Example usage:
const calculateTaxForIncome = calculateTax();
const income = 300000; // Assuming income is in Rupees
const taxAmount = calculateTaxForIncome(income);

console.log(`For an income of ₹${income}, the tax amount is
₹${taxAmount}`);
```

Q5. Write a JavaScript function called factorial that calculates the factorial of a non-negative integer using recursion. Test the function with different inputs.

Ans:- function factorial(value){

```
  if(value == 0)
  {
    return 1;
  }
  return factorial(value-1) * value;
```

```
}


console.log(factorial(5));


console.log(factorial(3));


console.log(factorial(10));
```

Q6. Write a JavaScript function called curry that takes a function as an argument and returns a curried version of that function. The curried function should accept arguments one at a time and return a new function until all arguments are provided. Then, it should execute the original function with all arguments.

Test the curry function with a function that adds two numbers.

Ans:-

```
function curry(func) {
  // Store the number of expected arguments for the original function
  const numArgs = func.length;

  // Define a recursive helper function to build up the arguments
  function curried(...args) {
    // If enough arguments are provided, execute the original function
    if (args.length >= numArgs) {
      return func(...args);
    } else {
      // If not enough arguments are provided, return a new curried function
      return (...moreArgs) => curried(...args, ...moreArgs);
    }
  }
  return curried;
}
// Test the curry function with a function that adds two numbers
function add(a, b) {
  return a + b;
}
const curriedAdd = curry(add);

// Test with partial application
const add5 = curriedAdd(5);
const result = add5(3); // Equivalent to add(5, 3)
console.log(result); // Output: 8
```

Complete