

## INTERMEDIATE LEVEL - FUNCTIONS FULL - ANSWERS

### SECTION 1 - Real-Time Function Logic

#### 1) Payroll System

```
function calculateSalary(basicSalary, bonusPercentage){
  let bonus = basicSalary * (bonusPercentage/100);
  let gross = basicSalary + bonus;
  let tax = gross * 0.05;
  let finalSalary = gross - tax;

  console.log("Basic:", basicSalary);
  console.log("Bonus:", bonus);
  console.log("Tax:", tax);
  console.log("Final Salary:", finalSalary);

  return finalSalary;
}
```

#### 2) Student Result System

```
function generateResult(name, marksArray){
  let total = marksArray.reduce((a,b)=>a+b,0);
  let avg = total / marksArray.length;
  let grade = avg>=80?"A":avg>=60?"B":avg>=40?"C":"Fail";
  return {name,total,average:avg,grade};
}
```

### SECTION 2 - Scope & Hoisting

#### 3) Output:

10

ReferenceError (b is not defined)

Reason:

var is function scoped, let is block scoped.

Fixed Code:

```
function demo(){
  let a,b;
  if(true){
    a=10;
    b=20;
  }
  console.log(a);
  console.log(b);
}
```

#### 4) Hoisting Output:

undefined

ReferenceError

Reason:

var is hoisted as undefined.

let is hoisted but in TDZ.

### SECTION 3 - Callback

#### 5) Order Processing

```
function generateInvoice(id){
  console.log("Invoice for",id);
}

function processOrder(orderId, callback){
  console.log("Order Processed");
  callback(orderId);
}
```

#### 6) Bank Transaction

```

let balance = 1000;
function sendSMS(msg){
  console.log("SMS:",msg);
}

function transaction(amount,type,callback){
  if(type==="deposit") balance+=amount;
  else if(type==="withdraw") balance-=amount;
  callback("Balance: "+balance);
}

```

#### SECTION 4 - Currying

##### 7) Price Builder

```

function priceBuilder(basePrice){
  return function(discount){
    return function(tax){
      let afterDiscount = basePrice - (basePrice*discount/100);
      return afterDiscount + (afterDiscount*tax/100);
    }
  }
}

```

#### SECTION 5 - IIFE

##### 8) Secure Module

```

const Company = (function(){
  let companyCode = "ABC123";
  return {
    getCompanyStatus: function(){
      return "Active";
    }
  }
})();

```

#### SECTION 6 - Generator

##### 9) Order ID Generator

```

function* orderIdGen(){
  let id=1001;
  while(true){
    yield "ORD"+id++;
  }
}

```

##### 10) Coupon Spin

```

function* coupon(){
  yield "10% OFF";
  yield "20% OFF";
  yield "50% OFF";
  yield "No Luck";
  yield "Jackpot";
}

```

#### SECTION 7 - Mini Project

```

const App = (function(){
  let cart=[];
  function* couponGen(){
    yield "10%";
    yield "20%";
  }
  function addToCart(p,price){ cart.push({p,price}); }
  function calculateTotal(){
    return cart.reduce((t,i)=>t+i.price,0);
  }
  function processPayment(total,callback){
    callback("Paid "+total);
  }
}

```

```
    }  
    return {addToCart,calculateTotal,processPayment,couponGen};  
  })();
```

Concept Answers:

- 1) Declaration is hoisted fully; expression is not.
- 2) Function that takes/returns another function.
- 3) Order ID generator.
- 4) To create private scope.
- 5) var=function scope, let/const=block scope; const cannot reassign.