

**Ghana Communication Technology University**

**Faculty of Computing and Information Systems**

**I.T 303 Java Programming**

**NAME**: MR. AWUSSAH, PAUL LARWEH  **LEVEL**: 300

**INDEX NUMBER**: 1711120394 **SESSION**: Evening

**2. A.**

public class Emolument {

// Encapsulated data fields

private double basicSalary;

private double taxRelief;

// Constructor to initialize Basic Salary and Tax Relief

public Emolument(double basicSalary, double taxRelief) {

this.basicSalary = basicSalary;

this.taxRelief = taxRelief;

}

// Method to get the basic salary

public double getBasicSalary() {

return basicSalary;

}

// Method to get the tax relief

public double getTaxRelief() {

return taxRelief;

}

// Method to compute SSNIT contribution (3.5% of Basic Salary)

public double SSNIT() {

return basicSalary \* 0.035;

}

// Method to compute Taxable Income

public double taxableIncome() {

return basicSalary - (taxRelief + SSNIT());

}

// Main method for testing the class

public static void main(String[] args) {

// Create an instance of Emolument

Emolument emolument = new Emolument(5000.0, 200.0);

// Display results

System.out.println("Basic Salary: " + emolument.getBasicSalary());

System.out.println("Tax Relief: " + emolument.getTaxRelief());

System.out.println("SSNIT Contribution: " + emolument.SSNIT());

System.out.println("Taxable Income: " + emolument.taxableIncome());

}

}

**2. B.**

public class Emolument {

// Encapsulated data fields

private double basicSalary;

private double taxRelief;

// Constructor to initialize Basic Salary and Tax Relief

public Emolument(double basicSalary, double taxRelief) {

this.basicSalary = basicSalary;

this.taxRelief = taxRelief;

}

// Method to get the basic salary

public double getBasicSalary() {

return basicSalary;

}

// Method to get the tax relief

public double getTaxRelief() {

return taxRelief;

}

// Method to compute SSNIT contribution (3.5% of Basic Salary)

public double SSNIT() {

return basicSalary \* 0.035;

}

// Method to compute Taxable Income

public double taxableIncome() {

return basicSalary - (taxRelief + SSNIT());

}

// Main method for testing the class

public static void main(String[] args) {

// Create an instance of Emolument

Emolument emolument = new Emolument(5000.0, 200.0);

// Display results

System.out.println("Basic Salary: " + emolument.getBasicSalary());

System.out.println("Tax Relief: " + emolument.getTaxRelief());

System.out.println("SSNIT Contribution: " + emolument.SSNIT());

System.out.println("Taxable Income: " + emolument.taxableIncome());

// Create an instance of MyEmolument

MyEmolument myEmolument = new MyEmolument(5000.0, 200.0);

// Display results for MyEmolument

System.out.println("Income Tax: " + myEmolument.incomeTax());

System.out.println("Total Deduction: " + myEmolument.totalDeduction());

System.out.println("Net Salary: " + myEmolument.netSalary());

}

}

class MyEmolument extends Emolument {

// Encapsulated data fields

private double basicSalary;

private double taxRelief;

// Non-arg constructor with default values

public MyEmolument() {

super(0, 0);

this.basicSalary = 0;

this.taxRelief = 0;

}

// Constructor to initialize Basic Salary and Tax Relief

public MyEmolument(double basicSalary, double taxRelief) {

super(basicSalary, taxRelief);

this.basicSalary = basicSalary;

this.taxRelief = taxRelief;

}

// Method to compute Income Tax

public double incomeTax() {

double taxableIncome = taxableIncome();

double incomeTax = 0;

if (taxableIncome <= 500) {

incomeTax = taxableIncome \* 0.05;

} else if (taxableIncome <= 1000) {

incomeTax = (500 \* 0.05) + ((taxableIncome - 500) \* 0.125);

} else {

incomeTax = (500 \* 0.05) + (500 \* 0.125) + ((taxableIncome - 1000) \* 0.175);

}

return incomeTax;

}

// Method to compute Total Deduction

public double totalDeduction() {

return SSNIT() + incomeTax();

}

// Method to compute Net Salary

public double netSalary() {

return getBasicSalary() - totalDeduction();

}

}

**2. C.**

public class Emolument {

// Encapsulated data fields

private double basicSalary;

private double taxRelief;

// Constructor to initialize Basic Salary and Tax Relief

public Emolument(double basicSalary, double taxRelief) {

this.basicSalary = basicSalary;

this.taxRelief = taxRelief;

}

// Method to get the basic salary

public double getBasicSalary() {

return basicSalary;

}

// Method to get the tax relief

public double getTaxRelief() {

return taxRelief;

}

// Method to compute SSNIT contribution (3.5% of Basic Salary)

public double SSNIT() {

return basicSalary \* 0.035;

}

// Method to compute Taxable Income

public double taxableIncome() {

return basicSalary - (taxRelief + SSNIT());

}

// Main method for testing the class

public static void main(String[] args) {

// Accept input from the user

String basicSalaryInput = JOptionPane.showInputDialog("Enter Basic Salary:");

String taxReliefInput = JOptionPane.showInputDialog("Enter Tax Relief:");

// Parse inputs to double

double basicSalary = Double.parseDouble(basicSalaryInput);

double taxRelief = Double.parseDouble(taxReliefInput);

// Create an instance of MyEmolument

MyEmolument staffSalary = new MyEmolument(basicSalary, taxRelief);

// Display results

StringBuilder result = new StringBuilder();

result.append("Basic Salary: ").append(staffSalary.getBasicSalary()).append("\n");

result.append("Tax Relief: ").append(staffSalary.getTaxRelief()).append("\n");

result.append("SSNIT Contribution: ").append(staffSalary.SSNIT()).append("\n");

result.append("Taxable Income: ").append(staffSalary.taxableIncome()).append("\n");

result.append("Income Tax: ").append(staffSalary.incomeTax()).append("\n");

result.append("Total Deduction: ").append(staffSalary.totalDeduction()).append("\n");

result.append("Net Salary: ").append(staffSalary.netSalary()).append("\n");

JOptionPane.showMessageDialog(null, result.toString(), "Staff Salary Details", JOptionPane.INFORMATION\_MESSAGE);

}

}

class MyEmolument extends Emolument {

// Encapsulated data fields

private double basicSalary;

private double taxRelief;

// Non-arg constructor with default values

public MyEmolument() {

super(0, 0);

this.basicSalary = 0;

this.taxRelief = 0;

}

// Constructor to initialize Basic Salary and Tax Relief

public MyEmolument(double basicSalary, double taxRelief) {

super(basicSalary, taxRelief);

this.basicSalary = basicSalary;

this.taxRelief = taxRelief;

}

// Method to compute Income Tax

public double incomeTax() {

double taxableIncome = taxableIncome();

double incomeTax = 0;

if (taxableIncome <= 500) {

incomeTax = taxableIncome \* 0.05;

} else if (taxableIncome <= 1000) {

incomeTax = (500 \* 0.05) + ((taxableIncome - 500) \* 0.125);

} else {

incomeTax = (500 \* 0.05) + (500 \* 0.125) + ((taxableIncome - 1000) \* 0.175);

}

return incomeTax;

}

// Method to compute Total Deduction

public double totalDeduction() {

return SSNIT() + incomeTax();

}

// Method to compute Net Salary

public double netSalary() {

return getBasicSalary() - totalDeduction();

}

}