Name: ____________________________

Student Id (last 4 digits) : ____________________________

• Write down the answers in the space provided. Use what you know from the lectures, labs, and the assigned readings.
• Please remember that the phrase “develop a function” means to design a function according to one of the recipes. You are not required to provide a template unless the problem specifically asks for one.
• You may obtain a maximum of 60 points.

Good luck.
1 Problem

This problem consists of several, independent parts. In order to solve one part, you do not need to know the solution for any other part.

1. Develop the function grossPay that computes the pay for an employee, based on the hours worked and the hourly pay rate.

```c
/* Compute the gross pay, given hours worked * and the hourly pay rate */
double grossPay(double hours, double rate){
    return (hours * rate);
}

/* Test the function grossPay */
void TestGrossPay(){
    testHeader("grossPay(double hours, double rate)");

    expected(150.00);
    actual(grossPay(10.0, 15.0));

    expected(480.00);
    actual(grossPay(40.0, 12.0));
}
```
2. Develop the function computeTax which computes the tax on the weekly income according to the following table:

<table>
<thead>
<tr>
<th>income</th>
<th>tax rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>under $200</td>
<td>10%</td>
</tr>
<tr>
<td>under $500</td>
<td>20%</td>
</tr>
<tr>
<td>$500 and over</td>
<td>30%</td>
</tr>
</tbody>
</table>

You must use a template here.

```c
/* Compute the tax on given income, using the tax table */
double computeTax(double income){
    /* Template:
       ... income < 200 ...
       ... income < 500 ...
       ... else ...
    */
    if (income < 200)
        return (income * 0.10);
    if (income < 500)
        return (income * 0.20);
    else
        return (income * 0.30);
}
```

/** Test the function computeTax */
void TestComputeTax(){
    testHeader("computeTax(double income)");
    expected(18.00);
    actual(computeTax(180.00));
    expected(46.00);
    actual(computeTax(230.00));
    expected(180.00);
    actual(computeTax(600.00));
}
3. Given these functions, develop the function `paycheck` which computes the net pay (i.e. gross pay minus the tax).

```c
/* Compute the net pay, given hours worked * and the hourly rate, using tax table */
double payCheck(double hours, double rate){
    return (grossPay(hours, rate) -
            computeTax(grossPay(hours, rate)));
}
```

```c
/****************************/
/* Test the function payCheck */
void TestPayCheck(){
    testHeader("payCheck(double hours, double rate)");

    expected(135.00);
    actual(payCheck(10.0, 15.0));

    expected(384.00);
    actual(payCheck(40.0, 12.0));

    expected(588.00);
    actual(payCheck(40.0, 21.0));
}
```
2 Problem

Design the class Tea, which represents a tea can in the grocery store. The relevant information is the brand name, the weight of the can, given in ounces, and the price of the can, given in cents.

class Tea {
    /*--------------------------------------------------------
     * The member data
     *--------------------------------------------------------*/
    // Brand name of the tea
    String name;

    // Weight in ounces
    double weight;

    // Price of the tea
    int price;

    /*--------------------------------------------------------
     * The constructor
     *--------------------------------------------------------*/
    Tea(String aName, double aWeight, int aPrice) {
        this.name = aName;
        this.weight = aWeight;
        this.price = aPrice;
    }

    /*--------------------------------------------------------
     * Define a Tea object and print the member data values.
     ********************************************************/
    void TestTea() {
        println("Define Tea objects, print member data values");
        println(new Tea("Earl Grey", 10.0, 300));
        println(new Tea("Jasmine", 8.0, 320));
    }
}
1. Develop the method `unitPrice`, which computes the price per ounce of this grocery item.

```java
/**
 * Purpose: to compute the unit price of this tea
 *
 * Template: ... this.name...this.weight ... this.price ...
 */
double unitPrice(){
    return this.price/this.weight;
}

/**
 * Test the method unitPrice.
 */
void TestUnitPrice(){
    Tea egt = new Tea("Earl Grey", 10.0, 300);
    Tea jas = new Tea("Jasmine", 8.0, 320);

    expected(30.0);
    actual (egt.unitPrice());

    expected(40.0);
    actual (jas.unitPrice());
}
```
2. Develop the method `isCheaperThan`, which determines whether the unit price is lower than some given price. Use a template.

```java
/*-----------------------------------------------
Purpose: to determine whether the unit price of this tea is less than some given price

Template:
...this.name...this.weight...this.price...this.unitPrice()
...aPrice
-----------------------------------------------

boolean isCheaperThan(int aPrice){
    return this.unitPrice() < aPrice;
}

/*********************************************************************************/
/*--------------------------------------------------------------------------------
Test the method isCheaperThan.
--------------------------------------------------------------------------------*/
void TestIsCheaperThan(){
    Tea egt = new Tea("Earl Grey", 10.0, 300);
    Tea jas = new Tea("Jasmine", 8.0, 320);

    expected(true);
    actual (egt.isCheaperThan(35));

    expected(false);
    actual (jas.isCheaperThan(35));
}
```
3. Draw the UML diagram for this class.
3 Problem

Given the UML diagram of classes Person and Address as shown:

1. Develop the method sameZip in the class Address that determines whether the address is in some zip code. Use template to develop this method.

The sameZip method.

```java
/* IN THE CLASS Address -------------------------------*/
/*--------------------------------------------------------*/
Purpose: to determine whether the zip code for this address is the same as the given zip code
---------------------------------------------------------*/

boolean sameZip(int aZip){
    return (this.zip == aZip);
}

/* IN THE TEST SUITE -----------------------------------*/
/********************************************************/
/*--------------------------------------------------------*/
Test the method sameZip.
---------------------------------------------------------*/

void TestSameZip(){
    Address grey = new Address("Houston", 30057);
    Address jas = new Address("Huntsville", 32054);

    expected(true);
    actual (grey.sameZip(30057));

    expected(false);
    actual (jas.sameZip(30057));
}
```
2. Develop the test suite that defines two objects in the class \texttt{Person} and performs two tests to determine whether either of these \texttt{Persons} lives in some zip code.

\begin{verbatim}
/*-----------------------------------------------
 Test to determine whether a person lives in the given zip code.
-----------------------------------------------*/
void TestZip(){

    Person earl =
        new Person("Earl Grey", new Address("Houston", 30057));

    Person jas =
        new Person("Jasmine Joy", new Address("Miami", 42054));

    expected(true);
    actual (earl.addr.sameZip(30057));

    expected(false);
    actual (jas.addr.sameZip(30057));
}
\end{verbatim}
4 Problem

Given a UML class hierarchy to represent items in the garden store, develop the classes Seedlings, Vegetables, and Flowers as specified in the UML diagram.