Part 1: Programming

A.

Implement a class ShoppingCart that models the shopping cart of an e-commerce site.

This shopping cart will be composed with an array of Item objects. You do not need to implement Item for this exam, only use it. Moreover, it is an abstraction and you do not need to know how it is implemented. Item has a getPrice() method that returns a float and a one-argument constructor that takes float that specifies the price.

ShoppingCart has the following constructors:

- A no-arg constructor.
- A single argument constructor that takes an array of Items.

Note: You should make use of the ‘this’ keyword (both of its applications).

ShoppingCart has the following variables:

- items: An array of Item objects representing the cart’s contents. The cart can hold a maximum of 10 Item objects (enforce this wherever necessary).
- numItems: An int that tracks the number of items in the users cart.
- taxRate: A double representing the tax rate to be applied, defaulted to .05. The value is the same for all objects of ShoppingCart (hint, hint!). The value could change and it should not be allowed to be negative.

Note: The numItems and taxRate variables should have accessor and mutator methods in addition to the methods listed below.

ShoppingCart has the following methods:

- addItem(i) Adds an item to the cart. Returns a boolean. If the cart is full, return false and do not add the item to the cart. Otherwise, add the item to the array and return true.
- subtotal(): Returns a float representing the combined cost of all the items in the cart not including tax.
- total(): Returns a float representing the subtotal plus tax. The tax amount is subtotal times the taxRate.

B.

Write a class called ‘TestShoppingCart’ with a main method that does the following:

1. Create a ShoppingCart
2. Adds 10 Items to it, each item should have a randomly generated price between $0 and $100.
3. Print the dollar amount of tax that the customer would have to pay.
Part 2: Code Reading

For code reading questions answer “When executed would the code display any output? If so what would it display? If not, why not?”.

A.

class Count {
  int count;
  public Count() {
    count = 0;
  }
}

public class Foo {
  public static void main(String[] args) {
    Count myCount = new Count();
    for (int i = 0; i < 100; i++) {
      increment(myCount);
    }
    System.out.println("myCount.count = " + myCount.count);
  }
  public static void increment(Count c) {
    c.count++;
  }
}

B.

class ThisConstructor {
  private int x;
  public ThisConstructor(String s) {
    this();
    System.out.println(s);
    System.out.println(" this ");
  }
  public ThisConstructor(int x) {
    this(" is ");
  }
  public ThisConstructor() {
    System.out.println(" a test ");
  }
  public static void main(String[] args) {
    int val = 2;
    ThisConstructor t = new ThisConstructor(val);
  }
}

C.

class Super {
  public String m() {
    return "Super";
  }
}

class Sub extends Super {
  public String m(String s) {
    return s + "Sub";
  }
}

public class Base {
  public static void main(String[] args) {
    Sub sub = new Sub();
    System.out.println(sub.m());
  }
}
Part 3: Short Answer

A. Draw a diagram showing what a stack data structure would contain after this sequence of operations.

    push('a'), push('b'), pop(), peek(), push('c'), push('d'), pop(), pop(), push('d'), peek()

B. How does the compiler determine which constructor you mean to call when you use the new operator on a class with multiple constructors?

C. Explain the following statement. “A variable of a reference type doesn’t actually contain an object of that type.”

Part 4: True/False

A. (1 point) A subclass can add behavior that is not present in the superclass.

B. (1 point) Instance variables should always be declared to be public.

C. (1 point) Whenever you use the new keyword, Java allocates space from a pool of memory called the static region.

D. (1 point) Method overloading is a technique where we replace a method of a superclass with one in a subclass.

E. (1 point) Object is the default superclass of every class, unless you specify a different superclass, then Object is not a superclass.