Homework M2: Generics

Due Feb 14 at 11:59pm **Points** 100 **Questions** 22
Available until May 30 at 11:59pm **Time Limit** None

Instructions

Review the **Homework FAQ** page. This page contains a video of how to use the provided tester file. **Make sure you know how to add the tester to your IDE and use it to test your code!** If you aren't sure how to use the tester or have any questions, post to the discussion board!

In this homework, you will:

- write code using generics
- evaluate code that uses generics

Homework Files

I have provided you with a driver/tester program that you can use to test your code before submitting. I **strongly** recommend running this program before you submit your homework. The program is designed to help you catch common errors and fix them before submitting.

The driver program has a series of *test methods* at the end. These are meant to streamline the running of multiple repeated tests. Don't worry too much if the test methods don't make sense- you don't need to do anything with these methods. They are just there to help with the test cases declared in the main method.

Important Note: The tester will not compile until the Trio class exists and has the required method headers. You can write the classes with placeholder code so the tester compiles. You can also comment out different parts of the tester while working on other parts.

Note: There are some tester lines you should un-comment and then see compiler errors. This is noted in the tester file.

TrioTester.java (https://ccsf.instructure.com/courses/47904/files/7281376/download?download_frd=1)

This quiz was locked May 30 at 11:59pm.

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	5,625 minutes	94 out of 100

Score for this quiz: **94** out of 100
Submitted Feb 12 at 1:27pm
This attempt took 5,625 minutes.

Question about Using Canvas

Question 1

6 / 6 pts

I want to make sure you know how to find the comments and feedback that are left on homework questions. So I left a "secret word" as feedback on Question 11 from Homework 1. Find that secret word and enter it here.

(Note: if you do not see the word on your Homework M1, I might not have had a chance to put it there yet. Check back in another day or two.)

Correct!

eureka

orrect Answers

eureka

Coding Questions

In the following questions, write a generic class called Trio. The Trio class is a data structure that holds an unordered group of three items.

Trio Characteristics

- Trios hold three items of the **same type**.
  - For example, a Trio could hold three Integers or three Strings or three Students, etc.
  - A Trio could **not** hold two Integers and a String.
- A Trio **can** contain duplicates.
- A Trio's items are **unordered.**
  - The order doesn't matter.
  - This is like a *set* in mathematics- sets are unordered. A Trio is different from a set, however, because a Trio can contain duplicates, while a set cannot.
  - For example, the Trio (3, 4, 5) is considered *the same as* the Trio (4, 5, 3) because they have the same elements. The Trio ("hi", "bye", "hello") is considered *the same as* the Trio ("hello", "hi", "bye") because they have the same elements. The order is ignored.

Class Header

The class header is:

```
public class Trio<T>
```

For full credit:

- follow **Java coding conventions (https://www.youtube.com/watch?v=va7zNueA9N0&list=PL5igFWijWBo1D0f-cJHH6jlJujtogqM_D&index=16)**
- follow general best practices and principles of object-oriented programming
- properly format and indent code
- follow naming conventions for variables, classes, and methods
- reduce duplicated code

Note: your Trio class **must** use generics in order to receive credit.

Question 2

3 / 5 pts

Write the complete class header and the instance data variables for the Trio class. Use generics.

For full credit:

- Use generics
- Declare variables in a way that supports encapsulation
- Follow naming conventions

Your Answer:

```
public class Trio {

        // instance data variables
        private T item1, item2, item3;
        private ArrayList itemArray;
}
```

-2 the class header needs to declare the generic type: public class Trio<T>

Question 3

10 / 10 pts

Write **two** constructors for the Trio class.

1. the first constructor takes in the three items as parameters
2. the second constructor takes in a single item and creates a Trio that consists of three of those items

For full credit:

Reduce duplicated code in the constructors and follow best practices for writing and invoking overloading constructors (discussed in the **M1**

video about writing classes (https://www.youtube.com/watch?v=X4W0OlHPEwU&list=PL5igFWijWBo1D0f-cJHH6jlJujtogqM_D&index=7)).

Your Answer:

```
// constructor 1
public Trio (T item1, T item2, T item3) {
        this.item1 = item1;
        this.item2 = item2;
        this.item3 = item3;
}

// constructor 2
public Trio(T item) {
        this(item, item, item);
}
```

Question 4

5 / 5 pts

Write getters and setters for each item in the Trio.

Your Answer:

```
// getter item1
public T getItem1() {
        return item1;
}

// setter item1
public void setItem1(T item1) {
        this.item1 = item1;
}

// getter item2
public T getItem2() {
        return item2;
}

// setter item2
public void setItem2(T item2) {
        this.item2 = item2;
}

// getter item3
public T getItem3() {
        return item3;
}

// setter item3
public void setItem3(T item3) {
```

```
        this.item3 = item3;
}
```

Question 5

5 / 5 pts

Write a **toString** method. The text representation of a Trio should include the three items contained in the Trio. The text should be formatted in a way that is human-readable (e.g., with spaces, tabs, or new lines).

Your Answer:

```
// toString method
@Override
public String toString() {
        return item1.toString() + "\t" + item2.toString() + "\t" + item3.to
String();
}
```

Question 6

8 / 8 pts

Write a method called **replaceAll**. The method takes in one item as a parameter and modifies the current Trio so that it holds three of that item.

Your Answer:

```
// replaceAll method
public void replaceAll(T item) {
        setItem1(item);
        setItem2(item);
        setItem3(item);
}
```

Question 7

8 / 8 pts

Write a method called **hasDuplicates** that returns true if at least two items within the Trio are equivalent to each other.

Note: this method determines duplicates based on whether items are **logically equivalent** (not whether items are aliases).

Your Answer:

```
// hasDuplicates method
// determines if there is a duplicate item within Trio instance
public boolean hasDuplicates() {
        if ((item1.equals(item2)) || (item1.equals(item3)) || (item2.equals
(item3))) {
                return true;
        }
        return false;
}
```

Question 8

8 / 8 pts

Write a method called **count** that takes in an item as a parameter and returns a count of how many times that item is in the Trio. Similar to the hasDuplicates method, this method counts based on logical equivalence.

Your Answer:

```
// getter ArrayList method
public ArrayList getItemArray(){
        itemArray = new ArrayList();
        itemArray.add(item1);
        itemArray.add(item2);
        itemArray.add(item3);

        return itemArray;
}

// count method
// counts how many of an item appears in Trio instance
public int count(T item) {
        // counter
        int itemCounter = 0;
```

```
        // loop through itemArray to find matching values
        for (int i = 0; i < getItemArray().size(); i++) {
                if (item.equals(getItemArray().get(i))) {
                        itemCounter++;
                }
        }
        return itemCounter;
}
```

consider if getItemArray should be a private helper method only accessible within the class; also, itemArray should not be an instance data variable; you correctly initialize it as a new list each time this method is called; since you do that, it should just be a local variable, not a variable that is kept as part of the class; use generics in the getItemArray method: public ArrayList<T> getItemArray()

Question 9

15 / 15 pts

Override the **equals** method from the Object class. The method returns true if the current Trio holds the same (logically equivalent) three items in any order as the Trio sent as a parameter and false otherwise.

Consider reviewing:

- the **M1 video about the equals method** (https://www.youtube.com/watch?v=gaj93xco_p4&list=PL5igFWijWBo1D0f-cJHH6jlJujtoggM_D&index=14) for non-generic classes (and the associated **practice example** (https://www.youtube.com/watch?v=e8r8W2xbLY4&list=PL5igFWijWBo1D0f-cJHH6jlJujtoggM_D&index=15))
- this week's lecture video **example** (https://youtu.be/mRQ2Z4bxn9w) , which discusses equals methods in generic classes.

Be sure to test your method with different cases, particularly cases where the Trios have duplicate items. (Use the provided tester file!)

For full credit:

- the method should ignore the order of the three elements

- the method should not alter the current Trio object or the Trio object passed in as a parameter
- the method should work correctly when either Trio holds duplicates;

Note: if using Trio<?> in your instanceof and cast statements gives you a compiler error, try using Trio<T>.

Your Answer:

```
// equals method
@Override
public boolean equals(Object obj) {
        if (obj instanceof Trio<?>) {
                // cast
                Trio otherTrio = (Trio) obj;

                // if both this and other Trio's have duplicates or both Tr
io's don't have duplicates
                // then proceed to compare items
                if ((this.hasDuplicates() && otherTrio.hasDuplicates()) ||
                                (!(this.hasDuplicates() && otherTrio.hasDup
licates()))) {

                        // variables to confirm a match
                        boolean firstMatch = false;
                        boolean secondMatch = false;
                        boolean thirdMatch = false;

                        // if both Trio's have the same item value and same
item count
                        for (int i = 0; i < (this.getItemArray().size());
i++) {
                                // variable for readability
                                T item = getItemArray().get(i);

                                for (T otherItem : otherTrio.getItemArray
()) {
                                        // first matching item
                                        if ((i == 0 && item.equals(otherIte
m)) &&
                                                        (count(item) == (ot
herTrio.count((otherItem))))) {
                                                firstMatch = true;
                                                continue;
                                        }
                                        // second matching item
                                        if ((i == 1 && item.equals(otherIte
m)) &&
                                                        (count(item) == (ot
herTrio.count((otherItem))))) {
                                                secondMatch = true;
                                                continue;
                                        }
                                        // third matching item
                                        if ((i == 2 && item.equals(otherIte
m)) &&
                                                        (count(item) == (ot
herTrio.count((otherItem))))) {
                                                thirdMatch = true;
                                                continue;
                                        }
```

```
                                }
                        }
                        // true if matches for all three items are found
                        if (firstMatch && secondMatch && thirdMatch) {
                                return true;
                        }
                }
        }
        return false;
}
```

you can simplify this by declaring the local variable using generics (Trio<T> otherTrio = (Trio<T>) obj; and then just directly comparing the counts: return count(item1)==otherTrio.count(item1) && count(item2)==otherTrio.count(item2) &&count(item3)==otherTrio.count(item3)

Multiple Choice Questions

Question 10

2 / 2 pts

Which of the following is true about the code below?

```
ArrayList<String> wordList = new ArrayList<String>();
wordList.add("hello");
wordList.add(4);
```

Correct!

- (selected) it has a syntax (compiler) error
- there is NO syntax or runtime error
- it will have a runtime error

Generics is used, so only Strings can be placed on the list.

Question 11

2 / 2 pts

Which of the following is true about the code below?

```
ArrayList<String> wordList = new ArrayList<String>();
wordList.add("hello");
String word = wordList.get(0);
```

- it has a syntax (compiler) error
- there is NO syntax or runtime error

Correct!

- it will have a runtime error

Question 12

2 / 2 pts

Which of the following is true about the code below?

```
ArrayList wordList = new ArrayList();
wordList.add("hello");
String word = wordList.get(0);
```

- there is NO syntax or runtime error

Correct!

- it has a syntax (compiler) error

it will have a runtime error

Generics is not used. (This is bad practice!) Because of that, retrieving objects from the list will return type Object, so those objects cannot be stored as other types.

Question 13

2 / 2 pts

Which of the following is true about the code below?

```
ArrayList wordList = new ArrayList();
wordList.add("hello");
wordList.add(4);
```

it has a syntax (compiler) error

Correct!

there is NO syntax or runtime error

it will have a runtime error

There is no error. But note that this code is bad practice! You should use generics with collection objects like lists.

Question 14

0 / 2 pts

Which of the following is true about the code below?

```
ArrayList list = new ArrayList();
list.add(4);
String s = (String) list.get(0);
```

ou Answered

- (●) it has a syntax (compiler) error

- ( ) there is NO syntax or runtime error

orrect Answer

- ( ) it will have a runtime error

Generics are not used. (This is bad practice!) The code will compile and allow you to add a number to the list. The compiler will let you retrieve an item and make the cast to store it as a String. But the item is not a String and so the code will crash at runtime.

Question 15

2 / 2 pts

Which of the following is true about the code below?

```
ArrayList<int> numberList = new ArrayList<>();
list.add(4);
```

- ( ) it will have a runtime error

Correct!

- (●) it has a syntax (compiler) error

- ( ) there is NO syntax or runtime error

You cannot use primitives as a generic type.

Question 16

2 / 2 pts

Which of the following is true about the code below?

```
public class MyClass<T> {

   private T thing;

   public MyClass() {
      thing = new T();
   }
}
```

- there is NO syntax or runtime error
- it has a syntax (compiler) error

Correct!

- it will have a runtime error

You cannot instantiate a variable of a generic type.

Question 17

0 / 2 pts

Which of the following is true about the code below?

```
public class MyClass<T> {

   public void method(T item) {
```

```
        System.out.println("hi!");
    }
}
```

ou Answered

- (●) it has a syntax (compiler) error
- ( ) it will have a runtime error

rrect Answer

- ( ) there is NO syntax or runtime error

Question 18

2 / 2 pts

Which of the following is true about the code below?

```
public class MyClass<T> {

   public void method(T item) {
      System.out.println(item.toString());
   }
}
```

Correct!

- (●) there is NO syntax or runtime error
- ( ) it has a syntax (compiler) error
- ( ) it will have a runtime error

The toString method is inherited from Object, so all object variables have this method.

Question 19

2 / 2 pts

Which of the following is true about the code below?

```
public class MyClass<T> {

   private T thing;

   public MyClass(T thing) {
      this.thing= thing;
   }

   public T method() {
      return thing;
   }
}
```

- it has a syntax (compiler) error
- it will have a runtime error

Correct!

- there is NO syntax or runtime error

Question 20

6 / 6 pts

Which of the following is true about each statement below? Assume T is a generic type.

Correct! **T[] array;** — there is NO syntax or run

Correct! **T[] array = new T[5];** — it has a syntax (compiler)

Correct!

T[] array = (T[]) new Object[5]; — there is NO syntax or run

Other Incorrect Match Options:

- it will have a runtime error

Question 21

3 / 3 pts

Does each of the following statements create an ArrayList that holds **only** Strings?

Correct!

ArrayList<String> list = new ArrayList<String>(); — yes- this creates an Arra

Correct!

ArrayList<String> list = new ArrayList<>(); — yes- this creates an Arra

Correct!

ArrayList<String> list = new ArrayList(); — no- there is a mistake in t

The <String> is required on the left. On the right, you can use <String> or <>. Leaving off the <> completely is a bug that can cause problems at runtime.

Question 22

1 / 1 pts

Which of the following is true about the code below?

```
public class MyClass<GENERICTYPE> {
}
```

Correct!

- (●) the code does NOT follow Java naming conventions
- ( ) the code follows Java naming conventions

By convention, generic types are single letters, such as T, S, E, K, V, etc.

Quiz Score: **94** out of 100