

CMSC 132 Quiz 2 Worksheet

The next quiz for the course will be on Wed, Sep 14. The following list provides additional information about the quiz:

- **Do not post any solutions to this worksheet in Piazza.**
- The quiz will be a written quiz (no computer).
- The quiz will be in lab session.
- Closed book, closed notes quiz.
- Answers must be neat and legible.
- Quiz instructions can be found at <http://www.cs.umd.edu/~nelson/classes/utilities/examRules.html>
- Make sure you know your section number and your TA's name.
- **You must take your quiz in your assigned lab/discussion section and not show up to a random discussion section. We will not grade quizzes taken in the incorrect section.**

The following exercises cover the material to be included in this quiz. Solutions to these exercises will not be provided, but you are welcome to discuss your solutions with the TAs or instructors during office hours. It is recommended that you try these exercises on paper first (without using a computer).

Exercises

1. Read the document available at:
<http://www.cs.umd.edu/class/fall2016/cmsc132-04XX/content/resources/officehoursinfo.html>
2. You should be prepare to write Junit tests. Be familiar with the material available at:
<http://www.cs.umd.edu/eclipse/JUnitTesting.html>
3. In this class, what would be the grade for a project submitted 25 hours after the deadline?
4. What is the good faith attempt?
5. Why people like pokemon go so much? ☺
6. The following prototypes will be used to answer this question:
 - i. `public int compute(double x)`
 - ii. `public void compute(double x)`
 - iii. `public void compute(int x)`
 - iv. `public void compute()`
 - v. `private void compute()`
 - a. Does method **i.** overload method **ii.**? Yes/No
 - b. Does method **i.** overload method **iii.**? Yes/No
 - c. Does method **i.** overload method **iv.**? Yes/No
 - d. Does method **iv.** overload method **v.**? Yes/No
 - e. Does method **ii.** overload method **iii.**? Yes/No
7. What is a better name for the method we call "constructor"?

8. How many constructors are associated with the following class?

```
public class Desk {
    private int edition;
    private String brand;

    public void setEdition(int edition) {
        this.edition = edition;
    }

    public void setBrand(String brand) {
        this.brand = brand;
    }

    @Override
    public String toString() {
        return "Desk [edition=" + edition + ", brand=" + brand + "];"
    }
}
```

9. The following two classes will be used for the questions below.

```
public class Newspaper {
    private int edition;
    private String name;

    public Newspaper(int edition, String name) {
        this.edition = edition;
        this.name = name;
    }

    @Override
    public String toString() {
        return "Newspaper [edition=" + edition + ", name=" + name + "];"
    }
}

public class ElectronicNewspaper extends Newspaper {
    private int articlesLimit;

    public ElectronicNewspaper() {
        articlesLimit = 100;
    }

    public ElectronicNewspaper(int articlesLimit) {
        super();
        this.articlesLimit = articlesLimit;
    }

    @Override
    public String toString() {
        return "ElectronicNewspaper [articlesLimit=" + articlesLimit + "];"
    }
}
```

- a. How many constructors are associated with the Newspaper class?
- b. Would the ElectronicNewspaper class compile?
- c. Why would you like to use the @Override annotation?

10. Given the classes below, indicate whether the assignments are valid or invalid. Notice that we are using two packages.

```
package toyPackage;
public class Toy {
    protected int size;
    static int max;
    public static final int temp = 10;
}

package experiment;
import toyPackage.*;
public class Driver {
    public static void main(String[] args) {
        Toy p = new Toy();

        p.size = 10; /* Valid or Invalid (Circle your choice) */
        p.max = 20; /* Valid or Invalid (Circle your choice) */
        Toy.max = 30; /* Valid or Invalid (Circle your choice) */
        Toy.temp = 40; /* Valid or Invalid (Circle your choice) */
    }
}
```

11. Define a class called Book with the specifications below.

a. Instance variables (all private)

- i. title – string
- ii. authorFirstName – string
- iii. authorLastName – string
- iv. year – integer

b. Methods (non-static unless specified otherwise)

- i. Constructor with parameters title, author's firstname, author's lastname, and year.
- ii. Default constructor that initializes the object with the values "NO_TITLE", "NO_FIRST_NAME", "NO_LAST_NAME", and -1.
- iii. Copy constructor.
- iv. equals method – two books are considered equal if they have the same title and author.
- v. compareTo method – we should be able to sort books based on the author's full name.
- vi. toString() – displays the values of instance variables.
- vii. Feel free to add any other method you understand is needed.

12. Define a class called ElectronicBook that extends the Book class above. The class has the following specifications:

a. Instance variables (all private)

- i. webAddress – string
- ii. sizeMB – int

b. Methods (non-static unless specified otherwise)

- i. Constructor with parameters title, author's firstname, author's lastname, year, web address, and sizeMB. This constructor must call the super class constructor.
- ii. Default constructor – Initializes the object with the values "NO_WEB_ADDRESS" and 0 for sizeMB. The values associated with the Book default constructor will be used.
- iii. Copy constructor
- iv. equals method
- v. toString – calls the super class toString method and displays the values for webAddress and sizeMB.
- vi. getWebAddress – returns the web address
- vii. compareTo method – we should be able to sort electronic books based on sizeMB.
- viii. Feel free to add any other method you understand is needed.

13. The **process** method has the signature below. The method prints each book and for electronic books it also prints the web address.

```
public static void process(ArrayList<Book> book);
```