بسم الله الرحمن الرحيم

KING FAHD UNIVERSITY OF PETROLEUM AND MINERALS

Information and Computer Science Department

2012/2013 Second Semester (Term 112)

ICS201: Introduction to Computing II (3-3-4)

MAJOR EXAM II

Thursday, April 19th 2012, 09:30 AM – 11:30 AM

120 MINUTES

Student Information

Name:									
ID:									
Section:									

Question No.	Maximum Score	Score
01	14	
02	10	
03	40	
04	38	
05	18	
TOTAL	120	

```
System.out.println("GoodLuck");
```

Question 1 (14 points):

I) (9 Points)

Select the most correct choice for each of the following multiple choice questions:

1) **Which of the following exceptions is subject to the "catch or declare" rule?**

   a. RunTimeException
   b. NullPointerException
   c. ArrayOutOfBoundException
   d. FileNotFoundException
   e. None of the above is correct

2) **In a try-catch-finally structure, finally block will get executed:**

   a. Only after the try block is executed successfully
   b. Only after one of catch blocks is executed (exception occurred)
   c. Will run anyway, regardless of an exception occurred or not
   d. Will run only if more than one exception occurred at once
   e. None of the above is correct

3) **In …… graphics, image is represented by continuous geometric objects, while in …… image is represented as a rectangular grid of colored pixels**

   a. vector, raster
   b. raster, vector
   c. jpg, gif
   d. gif, jpg
   e. None of the above is correct

II) (5 Points)

For the following method, determine the return value for each mentioned invocation:

```
int fun(int x){
   if (x == 3 || x == 4)
        return x - 1;
   else
        return x + fun(x + 1) + fun(x + 2);
}
```

fun(3) = 2

fun(2) = 7

fun(0) = 17

Question 2 (10 points):

I) (3 Points)

Define a checked exception class called *NotValidIntException*. The class should have the default message "Not a valid int". Exceptions of this class should also be capable of having user-defined custom messages.

See Attached Java file for Key Solution!

II) (7 Points)

Create a subclass of *JTextField* called *MyJTextField* and has a constructor that takes an *int* specifying the number of viewable characters on screen. This new subclass should have a *public int* method called *getInt()* that throws *NotValidIntException* defined above. In case the text field of an object of this subclass has a value that could be parsed as an *int*, then *getInt()* would return the parsed *int*, otherwise, it fires up an instance of *NotValidIntException* with the message: "Trimmed text is not int".

Hint: *Character* wrapper class has the method:

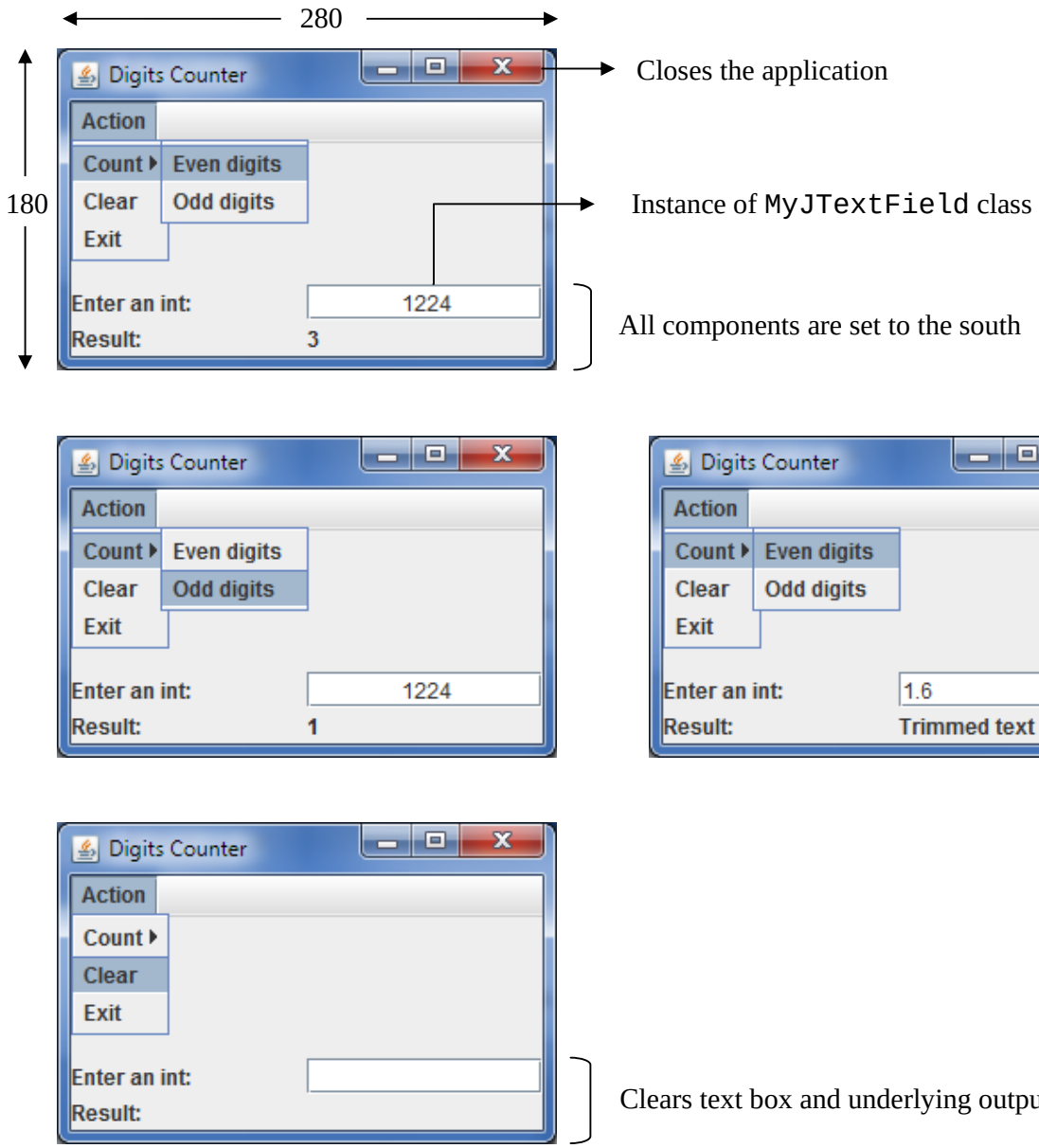
public static boolean isDigit(char ch)

This method will return *true* if *ch* is a digit (i.e. any value between '0' and '9'), otherwise, it returns *false*.

See Attached Java file for Key Solution!

Question 3 (40 points):

Design a GUI application EXACTLY as shown in the following snapshots:



Notes:

1. **Define a single class to hold the components, run the application and handle all events (all-in-one style).**
2. **Define the following recursive method:**

   public int countDigits(boolean oddCount, int n)

   Such that if *oddCount* is *true*, *countDigits(…)* will return the number of odd digits in *n* recursively, otherwise, it returns the number of even digits in *n* recursively.
3. **The input text box should be an instance of *MyJTextField* class (refer to the previous question for more detail).**
4. **In case "Even digits" or "Odd digits" menu item is clicked, the *getInt()* method of the text box should be called to parse the *int***

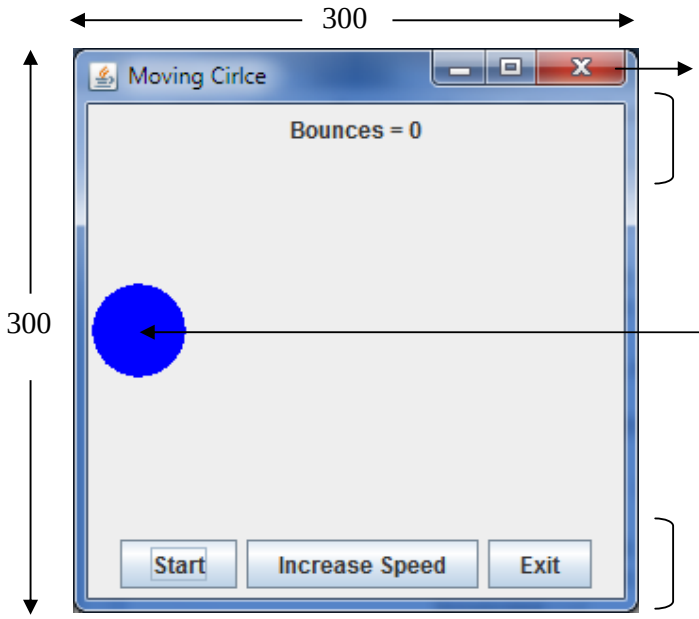
input. If the *int* input is parsed correctly, then *countDigits(…)* method should be called to get the number of even/odd digits, then result should be viewed in the output label (next to Result label). If the int input parsing caused an exception, the message of that exception should be viewed in the output label.

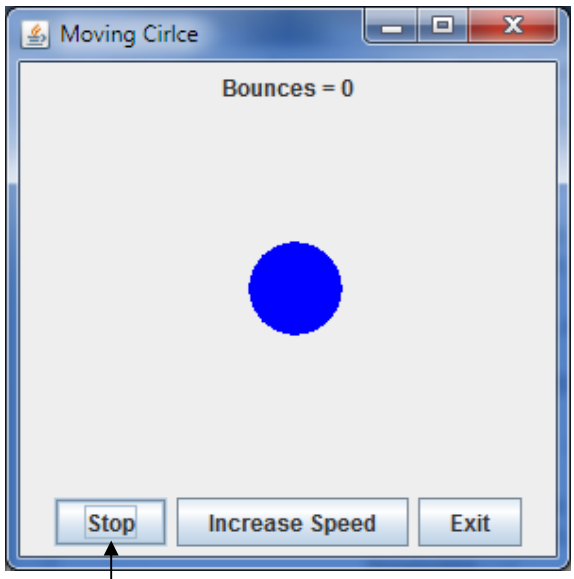
5. **The "Clear" menu item once clicked should set the input text box and output label to empty strings.**
6. **The "Exit" menu item and the X button (closing window button) once clicked should exit the application.**
7. **It is ok to use wildcards (*) in the import statements.**

See Attached Java file for Key Solution!

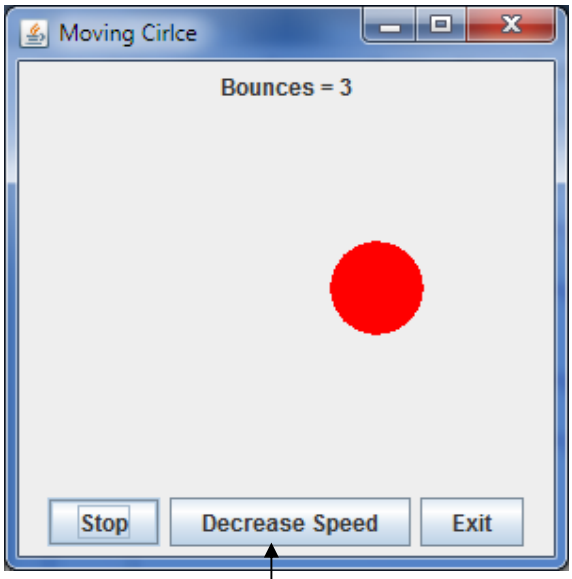
Question 4 (38 points):

Design a GUI application EXACTLY as shown in the following snapshots:





When clicked, circle moves, and text of the button changes to "Stop". Circle moves horizontally at 1 pixel per 10 milliseconds.



When clicked, button changes its text to "Decrease Speed" and circle moves horizontally at 1 pixel per 5 milliseconds.

Notes:

1. **Define a single class to hold the components, run the application, move the circle and handle all events (all-in-one style).**
2. **The label should keep track of all bounces.**
3. **When the widest pixel of the circle is 10 pixels away from each sides of the frame, it should bounce back and change its color.**
4. **Only the exit button closes the application.**

5. **The circle stops movement when the total number of bounces is equal to 5.**
6. **The "Start"/"Stop" button should stop or resume circle movement at any time when it is clicked. It also should change its text accordingly between "Start" and "Stop".**
7. **The "Increase Speed"/"Decrease Speed" button should increase or decrease circle movement at any time when it is clicked. It also should change its text accordingly.**
8. **It is ok to use wildcards (*) in the import statements.**

See Attached Java file for Key Solution!

Question 5 (18 points):

I) (12 Points)

Briefly list the steps needed to convert previous moving circle application into an applet. (No need to write code, just mention the steps)

See Attached Java file for Key Solution!

II) (6 Points)

Write a very short HTML document that would host and show the applet version of the moving circle application as close as possible to its original view.

See Attached Java file for Key Solution!