Name: [5] Yu-Ching Ho

CSC 204
Test 3
March 23, 2017

75

This is a closed book, closed notes, no computer, no calculator, no cell phone, test. Answer all of the questions in the space provided. Make sure that all of your answers are legible so that they can be graded. Points are indicated in [] and sum to 100. Enjoy!!

1. Complete the following Truth Table. Use "T" and "F" to indicate true and false: **[5]**

A	B	A && B	A || B	!A
T	T	T	T	F
T	F	F	T	F
F	T	F	T	T
F	F	F	F	T

✓

2. What is the output of the following Java code? **[5]**

```
for (int CSC = 100; CSC > 50;  CSC -= 25)
   for (int IST = 3; IST < 5; IST++)
      System.out.println(CSC + IST);
```

100 + 3
75 + 4

Output: 103 ✓ ✗
79 ✓

-2

3. What is the output of the following Java code? **[5]**

```
for (int CSC = 100; CSC > 50;  CSC -= 25)
   for (int IST = 3; IST < 5; IST++)
      System.out.println(CSC + "+" + IST);
```

Output: 100 + 3 ✓
75 + 4 ✓

more -2

-4

4. Write an entire public static method named "`isEven`" that is passed a single integer and returns true or false stating whether or not the integer is even. This method would return false when an odd number is passed in. **[15]**

```
public static void main (String[] args) {
    Scanner keyboard = new Scanner;
        int x = keyboard.nextInt;
        isEven(x);
}
public static int isEven(x) {
if(x%2 == 0){
    System.out.println("True");
}
else { System.out.println("False"); }}
```

5. Help me write an infinite loop that alternates printing "Orange" and "Black." Each underline needs some Java code. You should not write any more code than where you see underlines. **[5]**

```
int  turn = 0;

while (turn ≥ 0          )
{

if (  turn %2 == 0       )
   {
   System.out.println("Orange");

     turn ++;
   }
else
   {
   System.out.println("Black");

     turn ++;
   }
}
```

-5

6. Recall the Die class from our textbook. It simulates the cast of a single die. More specifically, when a Die object is constructed it is passed the desired number of sides for that die. A Die object can then call its "cast()" method which returns a random integer in the range of 1 to the number of sides. [10]

   The code from the Die class is listed below, except that the cast() method is left for you to write. Remember that the Random class has the method "`nextInt(int N)`".

```
import java.util.Random;
/**
   This class models a die that, when cast, lands on a randomface.
*/
public class Die
{
   private Random generator;
   private int sides;
   /**
      Constructs a die with a given number of sides.
      @param s the number of sides, e.g. 6 for a normal die
   */
   public Die(int s)
   {
      sides = s;
      generator = new Random();
   }
   /**
      cast()
      Simulates a throw of the die
      @return the face of the die
   */
```

```
public static int cast() {
   int faceOfDie = generator * sides;
   System.out.println("The face of the die is " + faceOfDie")
}
```

-5

```
}
```

/5

7. Simulate rolling a die a bunch of times to see how long it takes to get a specified die number. Assume you have the Die class available to use for this question. Write a public static method named "`rollCount`", that is passed an integer specifying what cast on a 6 sided die we are looking for, and returns how many casts it took to get that desired roll. [15]

```
public static int rollCount(){
  Scanner keyboard = new Scanner;
  System.out.println("Choose a number between 1 and 6");
        int xc = keyboard.nextInt;
        int rolls = 0;
while(faceOfDie != xc){
        rolls++;
  if(faceOfDie == xc){
    System.out.println("It took " + rolls + " rolls to get" + xc)}}}
```

-5

Cast??

8. Write a public static method named "`printStars`", that is passed an integer number of stars to be printed on a single line. This method should then print that number of "*"s, side by side on a single line, followed by a new line at the end. This method should not return anything, just print stars. [10]

```
Scanner keyboard = new Scanner;
System.out.println("Enter a number");
        int xc = keyboard.nextInt;

public static int printStars(xc){
for(int i=0; i<xc; i++){
  System.out.print("*");
}
System.out.println();
}
```

?

9. Assume your "`printStars`" method works as specified. Write a public static method named "`printTriangle`", that is passed a single integer for the width and height of a triangle, that calls your "`printStars`"method multiple times to produce a triangle as demonstrated below. [10]

```
public static int printTriangle(xc){
  for(int i=0; i<xc; i++){
    printStars(i);
  }
}
```

i+1

-1

```
printTriangle(3);
*
**
***

printTriangle(5);
*
**
***
****
*****
```

-6

10. The Picture API from our textbook can be found at the end of this test. It may help you remember some of the functionality of Pictures. [15]

Write a nested loop that transforms every pixel in t3Pic (given below) so that the picture is darker. Recall that the Color class stores colors as RGB values, and that RGB of (0,0,0) is black, and RGB of (255,255,255) is white. Note that lower numbers means darker colors.

getColorAt(int x, int
setColorAt(int x, int y, Color

Your strategy should be for each pixel:

a. Get the Color,
b. Calculate the RGB integer components,
   (remember the Color class has getRed(), getGreen(), and getBlue()),
c. Cut the RGB integer values in half,
d. Create a new color with these new RGB values,
e. Put the new color back into t3Pic.

```
int width = t3Pic.getWidth();
int height = t3Pic.getHeight();

for(int x = 0; x < width; x++)
   for(int y = 0; y < height; y++) {
   t3Pic.setColorAt( width, height, (t3Pic.getRed() --),
                                    (t3Pic.getGreen() --),
                                    (t3Pic.getBlue() --)
   if(t3Pic.getColorAt(width, height) == 0) {
      t3Pic.getRed() = 0;
      t3Pic.getGreen() = 0;
      t3Pic.getBlue() = 0;
   }
}
```

need to get ColorAt

x ? y

-5

?

BONUS: Name three other people in this class besides yourself and Dr. Allen. [5]

-5

Class Picture

Constructor Summary	
Picture() Constructs a blank picture.	

Method Summary	
void	**border(int width)** Adds a black border to the image.
Color	**getColorAt(int x, int y)** Gets the color of a pixel.
int	**getHeight()** Gets the height of this picture.
int	**getWidth()** Gets the width of this picture.
void	**load(String source)** Loads a picture from a given source.
void	**move(int dx, int dy)** Moves this picture by the given amount in x- and y-direction.
void	**pick()** Displays a file chooser for picking a picture.
void	**reload()** Reloads this picture, undoing any manipulations.
void	**scale(int newWidth, int newHeight)** Scales this picture to a new size.
void	**setColorAt(int x, int y, Color c)** Sets the color of a pixel.