# SO 1, PI 2, Elementary: Students are able to trace algorithms

## Question:

Draw the picture that will be produced by the following *\*paintComponent()\** method:  
  
public static void paintComponent(Graphics g) {  
 super.paintComponent(g);  
 g.drawLine(64, 10, 94, 32);  
 g.drawLine(94, 32, 82, 67);  
 g.drawLine(82, 67, 45, 67);  
 g.drawLine(45, 67, 64, 10);  
}

## Grade Distribution

Out of 10

9, 9, 9, 9, 8, 10, 10, 7, 10, 7, 10, 9, 10, 9, 10, 10, 10, 9

## Good Solution

(64, 10)

(94, 32)

(45, 67)

(82, 67)

## Medium Solution

## Poor Solution

# SO 2, PI 1, Elementary: Students are able to understand computing problems and define computing requirements

## Question:

## Grade Distribution

## Good Solution

## Medium Solution

## Poor Solution

# SO 2, PI 2, Elementary: Students are able to design solutions to computing problems using sound design principles and able to compare design options

## Question:

## Grade Distribution

## Good Solution

## Medium Solution

## Poor Solution

# SO 3, PI 1, Elementary: Students are able to implement computer-based and/or software systems using sound development principles

## Question:

## Grade Distribution

## Good Solution

## Medium Solution

## Poor Solution

# SO 3, PI 2, Elementary: Students are able to evaluate a computer-based and/or software system

## Question:

## Grade Distribution

## Good Solution

## Medium Solution

## Poor Solution

# SO 8, PI 1, Elementary: Students are able to use programming languages

Question:

Write a program that simulates rolling a pair of dice. The program will output a table similar to the following:

Average number of tries:

# : tries

2 : 40.30

3 : 18.80

4 : 12.29

5 : 9.11

6 : 7.10

7 : 7.19

8 : 7.44

9 : 10.69

10 : 10.79

11 : 19.50

12 : 34.25

The program should have a function that return the number of times it takes to roll two six sided dice to get the given number. To calculate the average number of tries your program should call the function several times and sum up the number of tries. Then divide by the number of times the function was called.

## Grade Distribution

Out of 10.

10, 10, 10, 5, 10 5, 7, 10, 10, 9, 10, 10, 7, 10 10, 0, 0, 5, 10, 10, 8, 0, 6, 10, 10, 7

## Good Solution

/\*\*

\* This program simulates rolling a pair of six sided dice and

\* calculates the average of how many rolls it took to get the

\* sum of the two dice to equal a given number.

\* @author

\*/

public class AverageRolls {

/\*\*

\* Does the experiment 1000 times and then prints the

\* average number of rolls it took to get a given number

\* @param args, the command argument

\*/

public static void main(String[] args) {

int dieNumber = 2; // Starting number to be rolled

int tries = 0; // Times it took to roll number during one experiment

int sum = 0; // Total number to times it took to roll number across all experiments

double average = 0.0; // Average to rolls it took to roll number

int totalTries = 0; // Total number of times to execute the experiment

// Print top of table

System.out.println("Average number of tries:");

System.out.println(" # : tries");

// Keep doing experiment until the number to roll is 12

do {

// While the total experiment is not 1000 times

while (totalTries != 1000) {

tries = Average(dieNumber); // Call subroutine Average and equal it to tries

sum += tries; // Increment sum by tries

totalTries++; // Increment totalTries by one

}

// Calculate average and print result

average = (double)sum / (double)totalTries;

System.out.println(" " + dieNumber + " : " + average);

dieNumber++; // Increment dieNumber by one

// Reset sum and totalTries for next dieNumber

sum = 0;

totalTries = 0;

} while (dieNumber <= 12);

}

/\*\*

\* Simulates rolling of two dice

\* @param dieNumber, Given dieNumber to roll

\* @return count number

\*/

public static int Average(int dieNumber) {

int dieOne; // Die number one.

int dieTwo; // Die number two.

int sum; // Sum of the two dice.

int count = 0; // Number of rolls.

// While the sum of the two dice doesn't equal to the given number

// Continue to roll the dice

do {

// Gets a random number generated from 1 to 6

dieOne = (int)(Math.random() \* 6) + 1;

dieTwo = (int)(Math.random() \* 6) + 1;

sum = dieOne + dieTwo; // Sum of both dice.

count++; // Increment the count by one roll

} while (sum != dieNumber);

// Return the count back to main

return count;

}

}

## Medium Solution

package hmw;

/\*\*

\* Program that simulates rolling a dice and calculates tries taken to get a number

\* @author

\*

\*/

public class H05 {

/\*\*

\* Subroutine that simulates rolling a dice and breaking when the dice equals user input

\* @param playGame

\* @return

\*/

public static int playGame(int num){

int dice1 = (int) (Math.random()\*6)+1;

int dice2 = (int) (Math.random()\*6)+1;

int dice = 0;

int count = 0;

while(dice != num){

dice = dice1 + dice2;

dice1 = (int) (Math.random()\*6)+1;

dice2 = (int) (Math.random()\*6)+1;

count++;

}

return count;

}

/\*\*

\* Subroutine that calculates average by dividing dice number by 1000 games

\* @param num

\* @return

\*/

public static int getAverage(int num){

int average = 0;

for(int i = 0; i < 1000; i++){

average += playGame(num);

}

return average/1000;

}

public static void main(String[] args) {

// TODO Auto-generated method stub

int a;

System.out.println("# : tries");

for(a = 2; a <=12; a++){

System.out.println(a + ":" + getAverage(a));

}

}

}

## Poor Solution

/\*\*

\*The program should have a function that return the number of times it takes to roll two six sided dice to get the given number.

\*To calculate the average number of tries your program should call the function several times and sum up the number of tries.

\*Then divide by the number of times the function was called.

\*/

import java.util.Scanner;

/\*\*

\* @author

\*

\*/

public class H05 {

public static boolean noNum = true;//creates a boolean variable for while loop

public static int count = 1;// creates count variable and sts to 1

public static int run = 0;// creates run variable and sets to 0

public static int sum = 0;//creates sum variable and sets to 0

public static int sides = 0;//creates sides variable and sets to 0

public static Scanner scan = new Scanner(System.in);//opens a scanner for user usage

public static void averaginator(){//creates a function to find the average rolls on a number

System.out.println("What number you want??");//asks user for desired number

int userNum = scan.nextInt();//scans user input

System.out.println("How many sides on your dice?");//asks user for desired size of dice

sides = scan.nextInt();//scans user input

System.out.println(userNum +" it is!! With " + sides +"sides!!");//declares users choice

System.out.println("Average number of tries: \n # : tries \n ");//creates the table

for(int i = 0; i < 1000; i++){//begins a for loop that runs 1000 times

while(noNum){//opens a while loop

int dice1 = (int) (Math.random() \* sides + 1);//random roll on dice

int dice2 = (int) (Math.random() \* sides + 1);//random roll on dice

if(dice1 + dice2 == userNum){//checks condition

noNum = false;//cancels the while loop

run++;//adds to the amount of times the average was found

}

else if(dice1 + dice2 != userNum){//checks condition

count++;//adds to count

}

}

System.out.println(run +": " + count);//prints initial results

sum += count;//begins to add the average

count = 1;//resets the count to 1

noNum = true;// turns the while loop back on

}

run = 0;//sets run amount back to 0

}

public static void main(String[] args){

int average; //creates a variable for average

while(true){//makes the program run forever

averaginator();//uses the fuction

average = sum/1000;//calculates the average

System.out.println("Your average for that number is " +average);//prints results

average = 0;//resets average to 0

sum = 0;// resets sum of results to 0

}

}

}

# SO 8, PI 2, Elementary: Students are able to use current development environments and frameworks.

## Question:

Write Java code for the following problems.

### Problem 1

Write a public static subroutine getString that asks the user to enter a hexidecimal String. Then uses getHexValue to convert that String to an int.

### Problem 2

Write a public static subroutine that calls public static int playGame(int num) and calculates the average score. The subroutine should take two int parameters num andnumberOfGames. numberOfGames is the number of time to call playGame.

## Grade Distribution

Out of 10.

10, 10, 9, 0, 8, 0, 5, 8, 5, 10, 7, 7, 7, 9, 10, 10, 0, 0, 0, 10, 7, 0, 0, 9, 10, 10, 10

## Good Solution

/\*\*

\* @author

\* 26 February 2016

\* Lab 12

\*/

public class HexValueClass {

/\*\*

\* main subroutine

\* In this case, it calls getHexValue, and prints the decimal value of

\* the test strings, and prints those decimal values to the user.

\*

\* calls getHexValue, which calls hexValue

\* @param args

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

}

/\*

\* Problem 1:

\* Takes a char, hexDigit, and returns a value, 0 through 15

\*

\* @param: char hexDigit

\* @return: int (the value of that hexidecimal digit)

\*/

public static int hexValue (char hexDigit) {

switch (hexDigit) {

case '0':

return 0;

case '1':

return 1;

case '2':

return 2;

case '3':

return 3;

case '4':

return 4;

case '5':

return 5;

case '6':

return 6;

case '7':

return 7;

case '8':

return 8;

case '9':

return 9;

case 'a':

case 'A':

return 10;

case 'b':

case 'B':

return 11;

case 'c':

case 'C':

return 12;

case 'd':

case 'D':

return 13;

case 'e':

case 'E':

return 14;

case 'f':

case 'F':

return 15;

default:

throw new IllegalArgumentException( hexDigit + "is not a legal hex character.");

}

}

/\*

\* Problem 2:

\* Takes a string, finds the decimal value of each character by calling

\* the other subroutine, hexValue. Returns the value of the hexidecimal string

\* as a whole in decimal.

\*

\* calls hexValue

\*

\* @param: String hexString (should be a string that consist of only 0-9,A-F)

\* @return: an int, the total value of that String in hexidecimal

\*/

public static int getHexValue (String hexString) {

int value = 0;

for (int i = 0; i < hexString.length(); i++ )

value = value\*16 + hexValue( hexString.charAt(i) );

return value;

}

}

## Medium Solution

/\*\*Write a public static subroutine hexValue that takes a char and returns the charís value as an int

\* Write a public static subroutine getHexValue that takes a String and calculates the Hexidecimal value fo the string. Use the hexValue subroutine

\*/

/\*\*

\* @author

\*

\*/

public class HexValueClass {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

System.out.println(hexValue('A'));

System.out.println(getHexValue("BEEF"));

System.out.println(getHexValue("FFFF"));

System.out.println(getHexValue("3A"));

}

public static int hexValue(char hexValue){

//subroutine to get the value of hexidecimal numbers

//converts from char to int

//switch statement with cases 0 through F

//each integer 0 through 9 returns that integer

//each char A through F returns the integer value for that char

switch (hexValue){

case '0':

return 0;

case '1':

return 1;

case '2':

return 2;

case '3':

return 3;

case '4':

return 4;

case '5':

return 5;

case '6':

return 6;

case '7':

return 7;

case '8':

return 8;

case '9':

return 9;

//hexidecimal cases begin

case 'A':

return 10;

case 'B':

return 11;

case 'C':

return 12;

case 'D':

return 13;

case 'E':

return 14;

case 'F':

return 15;

default: throw new IllegalArgumentException();

//will tell the user if there was a number of letter entered out side of 0 through F

}

}

public static int getHexValue (String hexValue){

//subroutine that calculates the hexidecimal value of a string of hex values

int value= 0;

//declare value as an int and give it a value of 0

int i;

//declare i as an int

for ( i = 0; i < hexValue.length(); i++ )

//i starts with value of 0, as long as it is less that the length of the hexValue value (0-F), it will increment by 1

value = value\*16 + hexValue( hexValue.charAt(i) );

//multiplies the value by 16

//adds the new value to hexValue

//hexValue is a character grabbed from the above subroutine

return value;

//returns the new evaluated value

}

}

## Poor Solution

/\*\*

\*

\*/

/\*\*

\* @author

\*

\*/

public class lab12 {

/\*\*

\* @param args

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

HexValueClass.gethexvalue("");//getting the subroutine gethexvalue from the class hexvalueclass

System.out.println(getString());//prints out and int from the subroutine getstring

}

public static int getString(){ //new subroutine getString

System.out.println("Enter a hexidecimal"); //asks user to print out a hexidecimal get string and converts into int

String user;

user = TextIO.getlnWord(); //getting a hexidecimal from the user

int dec;

dec = HexValueClass.gethexvalue(user); //dec is from the subroutine gethexvalue and what the user input

return(dec); //returns the decimal to the main subroutine

}}