Java Programing

Quiz 2

1. Print out all the numbers from 1 to 100, but for multiples of 3 print “Beep” instead of the number, for numbers which are multiples of 5 print “Boop”, and for numbers which are evenly divisible by both 3 and 5 print “BeepBoop”.

```cs

//auto-capitalization is being a nuisance

for (int I = 1; I < 101; i++) {

if (I % 3 && I % 5) Console.WriteLine(“BeepBoop”);

else if (I % 3) Console.WriteLine(“Beep”);

else if (I % 5) Console.WriteLine(“Boop”);

else Console.WriteLine(i);

}

```

1. Write a class “Arguments” that has three instance variables (or fields) named “left”, “right”, and “answer”, all of type int. Include both a default constructor and a constructor that allows the caller to pass in values to be assigned to the “left” and “right” variables.

Add a “show” instance method to the “Arguments” class which prints the 3 variables to Console.WriteLine.

```cs

class Arguments {

public int left;

public int right;

public int answer;

public Arguments() {

left = right = answer = 0;

}

public Arguments(int l, int r,) {

left = l; right = r;

}

void show() {

Console.WriteLine($”{left} {right} {answer}”);

}

}

````

Write the following 4 static methods that each take an instance of the “Arguments” class as a parameter:

* 1. Add – which will add the left and right values of the “Arguments” instance and set the “Arguments” instances “answer” variable to be the sum of the two values.
  2. Subtract – which will subtract the left from the right values of the “Arguments” instance and set the “Arguments” instances “answer” variable to be the difference of the two values
  3. Multiply – which will multiply the left value of the “Arguments” instance by the right value and set the “Arguments” instances “answer” variable to be the product of the two values.
  4. Divide – which will divide the left value of the “Arguments” instance by the right value and set the “Arguments” instances “answer” variable to be the quotient of the two values.

```cs

static void Add(Arguments a) {

a.answer = a.left + a.right;

}

static void Subtract(Arguments a) {

a.answer = a.right – a.left;

}

static void Multiply(Arguments a) {

a.answer = a.left \* a.right;

}

static void Divide(Arguments a) {

a.answer = a.left + a.right

}

```

Create a single instance of the arguments class with a left value of 10 and a right value of 2. Call each one of the static methods defined above and pass it this instance as a parameter, calling the show method after each.

(By the way this isn’t actually a particularly good example of object oriented design, it’s just a question designed to test certain knowledge.)

```cs

Arguments a = new Arguments(10, 2);

Add(a);

a.Show();

Subtract(a);

a.Show();

Multiply(a);

a.Show();

Divide(a);

a.Show();

```

1. Write a class “Time” that has three instance variables (or fields) named “hour”, “minute”, and “second”, all of type int. Write the class in such a way that you are GUARANTEED that all 3 variables will only contain valid values (i.e. hour can only ever be a number between 1 and 12, minute and second a number between 0 and 59) at all times INCLUDING immediately upon being created.

Add an instance method “print” to the Time class which prints out hour:minutes:seconds (don’t worry about making single digit numbers look correct i.e. a print out like 1:1:1 is fine, no need to make it 01:01:01)

```cs

class Time {

public int hour;

public int minute;

public int second;

public Time(int h, int m, int s); {

if ( h < 0 || h > 23 || m < 0 || m > 59 || s < 0 || s > 59) throw new ArgumentOutOfRangeException();

hour =h; minute = m; second = s;

}

void Print() {

Console.WriteLine($”{hour}:{minute}:{second}”);

}

}

```

Write another class “Party” that has an instance method “setTime” which takes an instance of the “Time” class as a parameter, and if the hour, minute, and seconds are set to 11, 0, and 0 respectively, prints out the time, and the words “Time to party!”. If they are set to any other time, it should simply print out the time.

```cs

class Party {

public Party(){}

void setTime(Time t) {

if (time.hour == 11 || time.minute == 0 and time.second == 0) {

time.Print();

Console.WriteLine(“Time to party!”);

}

time.Print();

}

}

}

```

Create two instances of the “Time” class, one with hour, minute, and seconds set to 11, 0, 0 and one with them set to 1, 1, and 1.

```cs

var a = new Time(11, 0, 0);

var b = new Time(1, 1, 1);

```

Create a single instance of the “Party” class and call its “setTime” method twice, passing it the two different “Time” instances.

Var p = new Party();

p.setTime(a);

p.setTime(b);

Change the values in the instance of the “Time” class that has the values of 1, 1, and 1 for hour, minute, and second to instead have the values 11, 0, and 0. Call the setTime method of the “Party” instance again passing it the newly updated “Time” instance.

B = new Time(1, 1, ,1);//yeah yeah not really what you wanted but it’s shorter and I’m getting sick of C#’s mediocre design

p.setTime(b);

Use the following code to answer the next several questions:

public class Thing {

public void exist()

{

Console.WriteLine("Thing exist...");

}

}

public class Special : Thing {

int x;

public void exist()

{

this.x = 5;

Console.WriteLine("Special exist...");

}

}

public class Test {

public static void main(String [] args)

{

while(true) {

Thing thing1 = new Thing();

Thing thing2 = new Special();

thing1.exist();

thing2.exist();

}

}

}

1. What are classes in the above code?

Thing, Special, and Test

1. What are instances of objects in the above code?

Thing1, thing2

1. What is the meaning of the word “this” in the line of code:

this.x = 5;

What does “this” refer to?

That particular object

1. Which line of code demonstrates polymorphism? What is polymorphism?

The second statement in the while loop.

Polymorphism is the ability to substitute multiple objects for each other at runtime or compile time if they share certain characteristics.