CS180

Exam 1 Review

What is the output to the following code?

```
System.out.println("2 + 2 = " + (2 + 2));
System.out.println("2 + 2 = " + 2 + 2);
```

What is the output to the following code?

```
System.out.println(String.valueOf(15+20));
System.out.println(String.valueOf("15+20"));
```

Evaluate the following values.

```
byte val1 = Byte.MIN_VALUE;
int val2 = Integer.MIN_VALUE - 1;
double val3 = 3 + 4;
int val4 = 3 / 4;
int val5 = 3 + 4.5;
```

What is the output to the following code?

```
int point = 15;
switch (point)
{
  case 0:
    System.out.println("point is 0");
  break;
    case 15:
  System.out.println("point is 15");
    case 30:
  System.out.println("point is 15 or 30");
    break;
  case 40:
    System.out.println("point is 40");
  default:
    System.out.println("Invalid point");
}
```

What is the output to the following code?

```
double rate = 1.5;
double price = 0.0;
if (rate >= 0.0 && rate < 1.0)
{
  price = 20 * rate;
}
else if (rate >= 1.0 && rate < 2.0)
{
  price = 15 * rate;
}
else if (rate >= 2.0)
{
  price = 10 * rate;
}
System.out.println(price);
```

What is the output to the following code?

```
int y = 100, x = 5;
while (y > 0)
{
  y--;
  if (y % x != 0)
  {
    continue;
  }
  System.out.println(y);
}
```

What is the output to the following code?

```
for (int k = 1, g = 4; k++ < g; --g)
  System.out.print(k + " " + g + " ");
```

What is the output to the following program?

```
public class ConsExample
{
  int value;

  public ConsExample(int v)
  {
    this.value = v;
  }

  public static void main(String args[])
  {
    ConsExample ce1 = new ConsExample(10);
    ConsExample ce2 = new ConsExample(10);
    if (ce1 == ce2)
      System.out.println("True");
    else
      System.out.println("False");
  }
}
```

What is the output to the following program?

```
public class ConsExample {
    public ConsExample()
    {
      System.out.println("A");
    }

    public void sample(String eg)
    {
      System.out.println(eg);
      ConsExample ce1 = new ConsExample();
      System.out.println("B");
    }

    public static void main(String args[])
    {
      ConsExample ce1 = new ConsExample();
      ce1.sample("C");
    }
}
```

What is the output to the following program?

```
public class ConsExample {
  int value;
  public ConsExample() {
    value = 10;
    System.out.println(value);
  }

  public void func1() {
    ConsExample ce2 = new ConsExample();
    ce2.value = 20;
    System.out.println(this.value);
    ce2.func2();
  }

  public void func2() {
    System.out.println(this.value);
  }

  public static void main(String args[]) {
    ConsExample ce1 = new ConsExample();
    ce1.value = 20;
    System.out.println(ce1.value);
    ce1.func1();
  }
}
```

What is the output to the following program?

```
public class Puzzle {
  private int x;
  public Puzzle() {
    x = 0;
  }
  public void f(int x, int y) {
    x = y++;
    this.x = x * ++y;
  }
  public void g(int x, int y) {
    x = y * y;
    y = this.x;
  }
  public int getX() {
    return x;
  }
  public static void main(String args[]) {
    Puzzle p = new Puzzle();
    int x = 5;
    int y = 6;
    p.f();
    p.g();
    System.out.println("x=" + x + ", y=" + y +
      ", p.getX()=" + p.getX());
  }
}
```

What are two examples of interfaces (the Java definition) in your every day life? Discuss how your examples are examples of interfaces.

What is the output to the following code snippets?

```
public interface Sleeper
{
  public void wakeUp();
}

public class TiredSleeper implements Sleeper
{
  public void wakeUp()
  {
    System.out.println("Ehhh…");
  }

  public void ignoreAlarm()
  {
    System.out.println("Go away…");
  }
}

public class RestedSleeper implements Sleeper
{
  public void wakeUp()
  {
    System.out.printn("Ready for the day!");
  }
}
```

```
Sleeper[] sleepers = new Sleeper[2];
sleepers[0] = new TiredSleeper();
sleepers[1] = new RestedSleeper();
for (int k=0; k<2; k++)
  sleepers[k].wakeUp();
```

```
TiredSleeper sleeper = new TiredSleeper();
sleeper.ignoreAlarm();
```

```
Sleeper sleeper = new TiredSleeper();
sleeper.ignoreAlarm();
```

Write a function which takes in an array of integers and returns true if each successive value in the array is at least twice the previous value in the array. For example
{2, 4, 9, 12} returns false (12 < 9*2)
{2, 4, 9, 25} returns true.

```
public boolean arrayDoublesUp(int[] nums)
{









}
```

Write a function which takes in an array of integers and returns true if each successive value in the array is at least twice the previous value in the array. For example
{2, 4, 9, 12} returns false (12 < 9*2)
{2, 4, 9, 25} returns true.

```
public boolean arrayDoublesUp(int[] nums)
{
  // check the argument for completeness, just do something reasonable
  if (nums == null)
    return false;

  for (int k=1; k<nums.length; k++)
  {
    if (nums[k] < 2*nums[k-1])
      return false;
  }
  return true;
}
```

Write a function which takes in two arrays – one a set of integers (*nums*) and the other a set of potential factors (*factors*) – and computes the number of integers in *nums* which has at least one factor in *factors*. For example, if:
nums = {2, 4, 9, 12, 15, 25}
factors = {3, 5}
Then numIntegersWithFactors(nums, factors) would return 4 (for the 9, 12, 15, and 25).

```
public int numIntegersWithFactors(int[] nums, int[] factors)
{









}
```

Write a function which takes in two arrays – one a set of integers (*nums*) and the other a set of potential factors (*factors*) – and computes the number of integers in *nums* which has at least one factor in *factors*. For example, if:

nums = {2, 4, 9, 12, 15, 25}

factors = {3, 5}

Then numIntegersWithFactors(nums, factors) would return 4 (for the 9, 12, 15, and 25).

```
public int numIntegersWithFactors(int[] nums, int[] factors)
{
  // check the arguments for completeness
  if (nums == null || factors == null)
    return 0;

  int numDivisible = 0;
  for (int k=0; k<nums.length; k++)
  {
    for (int i=0; i<factors.length; i++)
    {
      if (nums[k] % factors[i] == 0)
      {
        numDivisible++;
        break;
      }
    }
  }
  return numDivisible;
}
```