Name:_____

CS110 Introduction to Computing
Fall 2016
Practice Exam 2 Solution

This exam is closed note/closed book; computers are not permitted. Your work on this exam must be your own. Answer all questions in the space provided continuing on the back of page if necessary.

Assume that any given Processing statements do not have typographical errors.

In the code you write, focus more on getting it to exhibit the correct behavior instead of worrying about syntax. Partial credit will still be awarded if you have syntax errors.

Good Luck!

Question 1 Answer each of the following questions:

1. Consider the following Processing program fragment:

```
int[] data = {23, 45, 0, 17, 21, 78, 54, 19, 6, -4};
```

Using the values stored in **data**, fill in the following blanks:

a) **data[5]** = ___78_____
b) **data.length** = ___10_____

2. Declare an array, named **springColors,** to store a bunch (quantity unknown) of color values (**color**s):

```
color[] springColors;
```

3. Create the **springColors** array to store 15 color values:

```
springColors = new color[15];
```

4. Initialize the **springColors** array to contain all greens:

```
for(int i = 0; i < springColors.length; i++)
{
     springColors[i] = color(0, 255, 0);
}
```

5-9. The following questions are based on Processing statement below:

```
color    burlywood1 = color(255,211,155);
color    slateGrey  = color(112,128,144);
Building park       = new Building(burlywood1);
```

5. What is the name of the class being used? ____Building_____

6. What holds an instance of the object that was created? ____park_____

7. What is the name of the constructor? _____Building_____

8. What is the type of parameter used in the constructor? _____color_____

9. Create another instance in a variable called **dalton**, to be a `slateGrey Building` object:

```
Building dalton = new Building(slateGrey);
```

Question 2 Write a function **`countZero`** that takes an array of integers and returns the number of zeros in it.

```
int countZero(int[] nums)
{
      int zeroes = 0;
      for(int n : nums)
      {
            if(n == 0)
            {
                  zeroes++;
            }
      }
      return zeroes;
}
```

Question 3 Consider the following statements:

```
int[] items = {1, 2, 3, 4, 5, 6};
int sum = 0;

for (int i=items.length; i>items.length/2; i--) {
  items[i-1] = items[i-2];
  sum += items[i-1];
  println(sum);
}

for(int item : items)
{
  println(item);
}
```

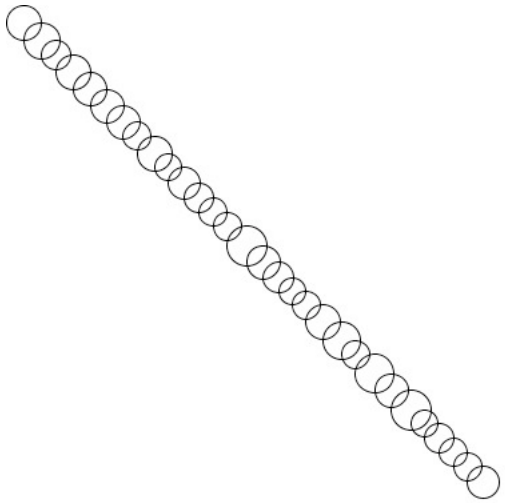
What will be printed? You may wish to make a table to show your work.

items[0]	items[1]	items[2]	items[3]	items[4]	items[5]	sum	i
1	2	3	4	5	6	0	
					5	5	6
				4		9	5
			3			12	4
							3

```
5
9
12
1
2
3
3
4
5
```

Question 4

1. Write a class called **`Ring`** that has four fields named **`x`**, **`y`**, **`s`**, and **`c`** that represent the **`Ring`**'s location (**`x`**,**`y`**), its size (**`s`**), and its color ©. The **`Ring`** constructor should initialize these fields using four arguments passed into it. The **`Ring`** class should also have a method called **`display()`** that draws the **`Ring`** centered at its x-y location as a simple unfilled ellipse with a 5 pixel border of the corresponding size (as a radius) and color.
2. Create an array and fill with 30 **`Ring`** objects with all color set to black and random sizes between 20 and 30.
3. Draw all the ring objects down the center diagonal of the sketch window, one over the next as shown below:



```
class Ring
{
     float x, y, s;
     color c;

     Ring(float ix, float iy, float is, color ic)
     {
          x = ix;
          y = iy;
          s = is;
          c = ic;
     }

     void display()
     {
          noFill();
          strokeWeight(5);
          stroke(c);
          ellipseMode(CENTER);
          ellipse(x, y, s * 2, s * 2);
     }
}

void setup()
{
     Ring[] rings = new Ring[30];

     float coord = 50;
     for(int i = 0; i < rings.length; i++)
     {
          float size = random(20, 30);
          coord += size;
          rings[i] = new Ring(coord, coord,
                              size, color(0, 0, 0));
     }

     for(Ring r : rings)
     {
          r.display();
     }
}
```

Question 5

Write a recursive function **`reverse`** that takes a **`String str`** and returns a reversed copy of it. For example, **`println(reverse("abc"));`** would print the string "cba". Note: The reverse function itself should not print anything.

```
String reverse(String str)
{
      if(str.length() == 0)
      {
            return "";
      }
      else
      {
            return reverse(str.substring(1)) +
                   str.substring(0, 1);
      }
}
```

Question 6

Write a function **`remove`** that takes an array of **`String`** called **`lst`** and an additional **`String str`**, returns the array with any occurrences of **`str`** replaced with **`null`**. If **`str`** does not appear in **`lst`**, then **`lst`** should be unchanged.

```
String[] remove(String[] lst, String str)
{
     for(int i = 0; i < lst.length; i++)
     {
          if(lst[i].equals(str))
          {
               lst[i] = null;
          }
     }
     return lst;
}
```

PROCESSING QUICK REFERENCE

Data Types
```
boolean, int, float, String
```

Conversion Functions
```
int(x), float(x), double(x)
```

Relational and Logical Operators

!= (inequality)	< (less than)	<= (less than or equal to),
== (equality)	> (greater than)	>= (greater than or equal to)
! (logical NOT)	&& (logical AND)	|| (logical OR)

Loops
```
for (init; test; update) {
  statements
}
```

```
while (expression) {
    statements
}
```

Conditionals

```
if (test) {
  statements
}
```

```
if (expression) {
  statements
}
else {
  statements
}
```

```
if (expression) {
  statements
}
else if (expression){
  statements
}
…
else {
  statements
}
```

Arrays
```
typeName[] arrayName = new typeName[N]
arrayName.length
```

Text Output
```
print(message), println(message)
```

Random
```
random(lower,upper)
```

File Input
```
loadStrings()
```

Working with Strings
```
split(value,delim), splitTokens(value,delim),
join(list,separator)
```