Question 1 (10 points) Write the correct answer in each of the following:

a) Write a Processing command to create a canvas of 400x300 pixels:

   \texttt{size(400, 300);}  

b) After the above command is carried out, what will be the value of the variable, \texttt{width}:

   \underline{400 \text{ pixels}}  

c) Write the command to translate the origin to the coordinates \textless 225, 50\textgreater :

   \texttt{translate(225, 50);}  

d) What will be printed by the following code segment:

   \begin{verbatim}
   int i = 1;
   while (i <= 8) {
       i++;
   }
   println(i);
   \end{verbatim}

   \underline{9 will be printed.}  

e) Consider the commands below:

   \begin{verbatim}
   int i=1;
   while (i < 12) {
       if (i\%3 == 0) {
           println(“Aloha”);
       }
       i = i + 1;
   }
   \end{verbatim}

   How many times would this code print the text “Aloha”? \underline{3}  

Question 1 (cont’d)
Suppose the function defined as:

   \texttt{void spongeBob(float x, float y, float s) {...
details deleted...}

   // spongeBob()
draws the Sponge Bob cartoon character with its top-left corner at \(<x, y>\) and of size \(s\) (i.e., its width and height are both \(s\)). Write the commands to do the following:

In the function definition above:

f) What is the name of the function defined above? ______spongeBob()_______

g) How many parameters does \textit{spongeBob()} take? _______3________

h) Use the function to draw a Sponge Bob of size 125x125 pixels with its top-left corner at \(<100, 50>\):

   \texttt{spongeBob(100, 50, 125);}

i) Use the function to draw a Sponge Bob of size 175x175 pixels with its top-left corner at \(<100, 150>\):

   \texttt{spongeBob(100, 150, 175);}

j) Use the function to draw an \textit{upside-down} Sponge Bob of size 100x100 pixels with its \textit{bottom-right} corner at \(<200, 240>\).

   \texttt{translate(200, 240);}
   \texttt{rotate(PI);}
   \texttt{spongeBob(0, 0, 100);}

\textbf{Question 2 (5 points)}
Write a conditional expression that would print “Just Right” whenever an integer variable \textit{temperature} has a value between 67 and 74, inclusive.

\begin{verbatim}
int temperature = ...;
// write your conditional here:
if (temperature >= 67 && temperature <= 74) {
   println(“Just Right”);
}
\end{verbatim}
Question 3 (5 points)
Rewrite the following using a while-loop:

```java
for (int i=1; i <= 15; i++) {
    if (i%3 == 0) {
        println("Hello");
    }
}

int i=1;
while (i <= 15) {
    if (i%3 == 0) {
        println("Hello");
    }
    i = i + 1;
}
```

Question 4 (10 points)
Write Processing code segment to draw a line from the center of a sketch to its border. It should be drawn tilted by a given angle, \theta, specified in radians. You may assume that the variable \theta is already defined. Your code should work for any size sketch window, and it is acceptable for the line to pass the border, but it must reach at least to the sketch border.

```java
float x = width * cos(theta);  // end point coordinates
float y = width * sin(theta);  // could use height

translate(width/2, height/2);  // Move to center of screen
line(0, 0, x, y);

Alternate:
float x1 = width/2;  // start point coordinates - sketch center
float y1 = height/2;
float x2 = x1 + width * cos(theta);  // end point coordinates
float y2 = y1 + width * sin(theta);  // could use height
line(x1, y1, x2, y2);
```
Question 5 (15 points)
The Processing program below is supposed to draw the following figure centered at every mouse click:

![Figure]

The width and height of the figure should always equal each other, and it should be a random number chosen between 50 and 150.

Complete the program below by writing the appropriate commands needed to accomplish the task. Continue on the back of page if more space is needed.

```java
void setup() {
    size(400, 400);
    background(255);
} // setup()

void draw() { }

void mousePressed() {
    float w = random(50, 150); // select a random size

    // Draw the black square
    fill(0);
    rectMode(CENTER);
    rect(mouseX, mouseY, w, w);

    // Draw the white circle
    fill(255);
    ellipse(mouseX, mouseY, w, w);
} // mousePressed()
```
Question 6 (15 points)

Write a complete Processing function that draws the following figure:

The function should be defined as shown below. The sketch shown was generated using the call:

drawSpokes(250, 250, 200, 20);

global void drawSpokes(int x, int y, int s, int n) {
    // draws n spokes centered at <x, y> of length s/2
    float r = s/2.0;
    float delta = TWO_PI/n;
    float theta = 0;

    pushMatrix();
    translate(x, y);
    for (int i=0; i < n; i++) {
        float x1 = r * cos(theta);
        float y1 = r * sin(theta);
        line(0, 0, x1, y1);
        theta += delta;
    }
    popMatrix();
} // drawSpokes()

Alternate version, using rotation…

global void drawSpokes(int x, int y, int s, int n) {
    float r = s/2.0;
    float delta = TWO_PI/n;

    pushMatrix();
    translate(x, y);
    for (int i=0; i < n; i++) {
        line(0, 0, r, 0);
        rotate(delta);
    }
    popMatrix();
} // drawSpokes()
Question 7 (10 points)
Consider the following code segment. Using the table below, trace the code segment to determine what will be printed by the call to `mystery(3, 8):

```java
void mystery(int a, int b) {
    int x = a;
    for(int i = 0; i < b - a; i++) {
        println(x);
        if(i % 2 == 0) {
            x = b - i;
        } else {
            x = a + i;
        }
    }
    println(x);
} // mystery()
```

Write what is printed here:

```
3
8
4
6
6
4
```

Question 8 (15 points)
Write a Processing function `sum(int n)` that computes and returns the summation:

\[1^2 + 3^2 + 5^2 + \ldots + n^2\]

```java
int sum(int n) {
    int result = 0;
    for (int i = 1; i <= n; i = i + 2) {
        result = result + i;
    }
    return result;
} // sum()
```
Question 9 (15 points)
Write a Processing function isPrime(int n) that detects whether a number n is prime. A prime number has no even divisors except for 1 and the number itself (see algorithm below). The function returns true if the number, n is prime, false otherwise. Do not worry about what happens if the parameter is less than 2.

To see if a number N is prime do the following
for each number (call it i) starting at 2 and counting up to (N-1)
    if i divides N evenly then N is not a prime number
otherwise, N is prime

boolean isPrime(int n) {
    for (int i = 2; i < n; i++) {
        if (n % i == 0) {
            return false;
        }
    }
    return true;
} // isPrime()