1 Short Answer (15 Points Each)

1. Write a method called `RollCount` that takes in two integer parameters `rolls` and `target`. The method should simulate the rolling of two die, `rolls` times, and count the number of rolls that result in the target value. The method should return that count.

   Solution:

```
public static int RollCount(int rolls, int target) {
    int count = 0;
    for (int i = 0; i < rolls; i++) {
        int die1 = (int) (Math.random() * 6) + 1;
        int die2 = (int) (Math.random() * 6) + 1;
        if (die1 + die2 == target)
            count++;
    }
    return count;
}
```

2. Write a method called `ExtractFirst` that takes a single string as a parameter, extracts the first word from the string and returns that word.

   Solution:

```
public static String ExtractFirst(String s) {
    int pos = s.indexOf(" ");
    String sub = s.substring(0, pos);
    return sub;
}
```

3. Write a method called `InputIntRange` that takes in two integer parameters `min` and `max`. The method should continually ask the user to input an integer value until the input value is between `min` and `max` (inclusively). Once the input is in the range the method should return that value.

   Solution:

```
public static int InputIntRange(int min, int max) {
    Scanner kb = new Scanner(System.in);
    int value = 0;
    do {
        System.out.print("Input an integer: ");
        value = kb.nextInt();
    } while (value < min || value > max);
    return value;
}
```

4. Write a method called `DoubleFactorial` that takes a single integer parameter `n` and returns the double factorial of `n`. The double factorial of a number n is defined to be

$$n!! = n \cdot (n-2) \cdot (n-4) \cdots 1$$

   We define $0!! = 1$ and if the value of n is less than 0 simply have the method return -1. For example, $4!! = 4 \cdot 2 = 8$, $5!! = 5 \cdot 3 \cdot 1 = 15$, $6!! = 6 \cdot 4 \cdot 2 = 48$, and $10!! = 10 \cdot 8 \cdot 6 \cdot 4 \cdot 2 = 3840$.

   Solution:

```
public static int DoubleFactorial(int n) {
    if (n < 0)
        return -1;

    int df = 1;
    for (int i = n; i > 0; i-=2)
        df *= i;
    return df;
}
```

2 Program Traces (15 Points Each)

1. For each of the given inputs, write the output of the program.

```
1  import java.util.Scanner;
2
3  public class Exam2Trace1 {
4
5      public static int mth1(int b, int c, int a) {
6          System.out.println("In Method 1");
7          System.out.println(a + " " + b + " " + c);
8          return a - b * c;
9      }
10
11     public static int mth2(int a, int b, int c) {
12         System.out.println("In Method 2");
13         System.out.println(a + " " + b + " " + c);
14         if (a > b)
15             return mth3(a, b, c);
16         else
17             return mth3(b, a, c);
18     }
19
20     public static int mth3(int c, int b, int a) {
21         System.out.println("In Method 3");
22         System.out.println(a + " " + b + " " + c);
23         return mth1(a, b, c);
24     }
25
26     public static int mth4(int b, int a, int c) {
27         System.out.println("In Method 4");
28         System.out.println(a + " " + b + " " + c);
29         if (a > b && b > c)
30             return a;
31         else if (a > b)
32             return c;
33         else
34             return mth2(c, b, a);
35     }
36
37     public static void main(String[] args) {
38         Scanner kb = new Scanner(System.in);
39         System.out.print("Input: ");
40         int a = kb.nextInt();
41         int b = kb.nextInt();
42         int c = kb.nextInt();
43
44         System.out.println(mth1(a, b, c));
45         System.out.println();
46         System.out.println(mth2(a, b, c));
47         System.out.println();
48         System.out.println(mth4(a, b, c));
49     }
50 }
```

(a) `Input: 1 2 3`

Solution:

```
In Method 1
3 1 2
1

In Method 2
1 2 3
In Method 3
3 1 2
In Method 1
2 3 1
-1

In Method 4
2 1 3
3
```

(b) `Input: 3 2 1`

Solution:

```
In Method 1
1 3 2
-5

In Method 2
3 2 1
In Method 3
1 2 3
In Method 1
3 1 2
1

In Method 4
2 3 1
In Method 2
1 3 2
In Method 3
2 1 3
In Method 1
3 2 1
1
```

2. For each of the given inputs, write the output of the program.

```
1  import java.util.Scanner;
2
3  public class Exam2Trace2 {
4
5      public static String DoSomething(String str1, String str2, int p) {
6          str1 += " ";
7          int c = 0;
8          int pos = -1;
9          while (c < p) {
10             pos = str1.indexOf(str2, pos + 1);
11             if (pos >= 0)
12                 c++;
13             else
14                 return "Error";
15         }
16         c = pos;
17         while (str1.charAt(c) != ' ') {
18             c--;
19         }
20         c++;
21         pos = str1.indexOf(" ", c);
22         return str1.substring(c, pos);
23     }
24
25     public static void main(String[] args) {
26         Scanner kb = new Scanner(System.in);
27         System.out.print("Input String: ");
28         String s1 = kb.nextLine();
29         System.out.print("Input String: ");
30         String s2 = kb.nextLine();
31         System.out.print("Input Number: ");
32         int a = kb.nextInt();
33         System.out.print(DoSomething(s1, s2, a));
34     }
35 }
```

(a) `Input String: Methods are also known as functions and subroutines.`
`Input String: o`
`Input Number: 4`

Solution:

`functions`

Values of `pos` in the loop: 4, 15, 19, 32

Final value of `c`: 26

(b) `Input String: Methods are also known as functions and subroutines.`
`Input String: s`
`Input Number: 2`

Solution:

`also`

Values of `pos` in the loop: 6, 14

Final value of `c`: 12

3 Coding (20 Points)

Do one and only one of the following exercises.

1. Write a program that will simulate rolling 2 die repeatedly until you get a run of face values from 2 up to a given number. That is, a run of 2 would be rolling a 2, then 3, a run of 3 would be rolling a 2, 3, 4 consecutively, a run of 4 would be rolling 2, 3, 4, 5 consecutively, and so on. Have the program count the number of rolls needed for each possible run. The output should look like the following.

```
Number of rolls needed for a run from 2 to 3 = 353
Number of rolls needed for a run from 2 to 4 = 8326
Number of rolls needed for a run from 2 to 5 = 17587
Number of rolls needed for a run from 2 to 6 = 217515
Number of rolls needed for a run from 2 to 7 = 520554
Number of rolls needed for a run from 2 to 8 = 11370125
Number of rolls needed for a run from 2 to 9 = 231679566
Number of rolls needed for a run from 2 to 10 = 3163916395
Number of rolls needed for a run from 2 to 11 = 67186538451
Number of rolls needed for a run from 2 to 12 = 722168518658
```

Solution:

```
1  public class Exam2Program1 {
2
3      public static void main(String[] args) {
4          for (int run = 3; run <= 12; run++) {
5              boolean done = false;
6              int target = 2;
7              long count = 0;
8              while (!done) {
9                  int die1 = (int) (Math.random() * 6) + 1;
10                 int die2 = (int) (Math.random() * 6) + 1;
11                 int roll = die1 + die2;
12
13                 if (roll == target)
14                     target++;
15                 else
16                     target = 2;
17
18                 count++;
19                 if (target == run + 1)
20                     done = true;
21             }
22
23             System.out.println("Number of rolls needed for a run from 2 to " + run + " = " + count);
24         }
25     }
26 }
```

2. Write a program that will take an input string from the user and count the number of vowels in each word. The main program should take the input string and extract each word of the string, one by one. It should then call the a method that takes in a string, assumed to be a single word, counts the number of vowels and returns that number to the main. The main program should also print out the word and vowel count. The output should look like the following.

```
Input String: Methods are also known as functions and subroutines
Word: Methods    Count = 2
Word: are    Count = 2
Word: also    Count = 2
Word: known    Count = 1
Word: as    Count = 1
Word: functions    Count = 3
Word: and    Count = 1
Word: subroutines    Count = 5
```

Solution:

```
1  import java.util.Scanner;
2
3  public class Exam2Program1 {
4
5      public static int CountVowels(String s) {
6          int count = 0;
7          s = s.toLowerCase();
8          for (int i = 0; i < s.length(); i++) {
9              char ch = s.charAt(i);
10             if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u')
11                 count++;
12         }
13         return count;
14     }
15
16     public static void main(String[] args) {
17         Scanner kb = new Scanner(System.in);
18         System.out.print("Input String: ");
19         String s = kb.nextLine();
20         s = s + " ";
21         int pos = 0;
22         while (pos >= 0) {
23             int start = pos;
24             pos = s.indexOf(" ", pos + 1);
25             if (pos >= 0) {
26                 String word = s.substring(start, pos);
27                 word = word.trim();
28                 int count = CountVowels(word);
29                 System.out.println("Word: " + word + "    Count = " + count);
30             }
31         }
32     }
33 }
```