Question 1

[20 marks]

1 Write a Java program given the following specification and provide comments which explain how your algorithm works.

Problem Statement
The goal is to read in a number and to output the nearest prime.
If two primes are equidistant then output the lower one.

Input Format
An integer *N*.

Output Format
The closest prime to *N*.

Constraints
2≤*N*≤10000

Sample Input
3856

Sample Output
3853

```java
import java.util.Scanner;

public class Q1 {
    public static void main (String args[]) {
        Scanner sc = new Scanner(System.in);
        int inputNum = sc.nextInt();
        sc.close();

        int nearestPrime = findNearestPrime(inputNum);
        System.out.println(nearestPrime);
    }

    // Find nearest Prime
    public static int findNearestPrime(int input) {
        int lowerPrime = input;
        int upperPrime = input;
        while(!isPrime(lowerPrime)) lowerPrime--;
        while(!isPrime(upperPrime)) upperPrime++;
```

CS210-2016-January

```
        if(Math.abs(lowerPrime - input) > Math.abs(lowerPrime -input)) {
            return upperPrime;
        }
        // If two primes are equidistant then output the lower one.
        else  {
            return lowerPrime;
        }
    }

    //Check whether the number is Prime
    public static boolean isPrime(int number) {
        if (number <= 1) return false;

        for(int i = 2; i < number; i++) {
            if(number % i == 0) return false;
        }

        return true;
    }
}
```

CS210-2016-January

Question 2

[20 marks]

2 Write a Java program given the following specification and provide comments which explain how your algorithm works.

Problem Statement
The goal is to sort a list of words in reverse alphabetical order.

Input Format
The first line contains *N*, the number of words to be sorted, followed by a line with *N* words, each separated by a space.

Output Format
A line consisting of the words sorted in reverse alphabetical order, each separated by a space.

Constraints
$1 \leq N \leq 100$

Sample Input
3
one two three

Sample Output
two three one

```java
import java.util.LinkedList;
import java.util.Collections;
import java.util.Scanner;

public class Q2 {
   public static void main (String args[]) {
      Scanner sc = new Scanner(System.in);
      LinkedList<String> list = new LinkedList<String>();

      // input the number of words to be sorted
      int N = Integer.parseInt(sc.nextLine());
      // input a line with N words, and put it in List

      for(int i = 0; i < N; i++) {
         String inputLine = sc.nextLine();
         list.add(inputLine);
      }
      sc.close();
```

CS210-2016-January

```
      // Sort in reverse alphabetical order.
      Collections.sort(list, Collections.reverseOrder());

      // Print out the sorted words
      for(String s : list) {
         System.out.println(s);
      }
   }
}
```

CS210-2016-January

Question 3

[20 marks]

3 Write a Java program given the following specification and provide comments which explain how your algorithm works.

Problem Statement
Use a stack to check if a sentence is a palindrome or not. You must write your own Stack class. A palindromic sentence is one that reads the same forwards as backwards when you ignore all the spaces. Upper and lower case letters should be treated as equivalent. If the sentence is palindromic, output TRUE, otherwise output FALSE.

Input Format
An input string *S*.

Output Format
Either TRUE if *S* is a palindrome or FALSE otherwise.

Constraints
1≤length(*S*)≤100

Sample Input
Ten animals I slam in a net

Sample Output
TRUE

Answer 1 – Interface

```
import java.util.Stack;
import java.util.Scanner;

public class Q3_Interface {
   public static void main (String args[]) {
      Scanner sc = new Scanner (System.in);
      String sentence = sc.nextLine();
      sc.close();
      if(isPalindrome(sentence)) {
         System.out.println("TRUE");
      }
      else {
         System.out.println("FALSE");
      }
   }
```

CS210-2016-January

```
   // Check whether the String is Palindrome
   public static boolean isPalindrome(String input) {
      Stack<Character> s = new Stack<Character>();

      // Convert String to char Array, all lower case.
      char []charArray = input.toLowerCase().toCharArray();

      // Input all char from first to last
      for (char c: charArray) s.push(c);

      // Check char from last to first
      for (char c: charArray) {
         if(c != s.pop()) return false;
      }
      return true;
   }
}
```

Answer 2 – Full Stack Class

```
import java.util.Scanner;

public class Q3_FullStackClass {
   public static void main (String args[]) {
      Scanner sc = new Scanner (System.in);
      String sentence = sc.nextLine();
      sc.close();
      if(isPalindrome(sentence)) {
         System.out.println("TRUE");
      }
      else {
         System.out.println("FALSE");
      }
   }
```

CS210-2016-January

```
   // Check whether the String is Palindrome
   public static boolean isPalindrome(String input) {
      FullStack s = new FullStack(100);

      // Convert String to char Array, all lower case.
      char []charArray = input.toLowerCase().toCharArray();

      // Input all char from first to last
      for (char c: charArray) s.push(c);

      // Check char from last to first
      for (char c: charArray) {
         if(c != s.pop()) return false;
      }

      return true;
   }
}

class FullStack{
   private int maxSize; // size of stack array
   private char[] stackArray;
   private int top; // top of stack

   public FullStack(int s) { // constructor
      maxSize = s; // set array size
      stackArray = new char[maxSize]; // create array
      top = -1; // no items yet
   }

   public void push(char j) { // nput item on top of stack
      top++;
      stackArray[top] = j; // increment top, insert item
   }

   public char pop() { // take item from top of stack
      return stackArray[top--]; //access item, decrement top
   }
}
```

CS210-2016-January

Question 4

Question a

[20 marks]

4 a) Identify the output that the following Java code produces and explain your reasoning clearly. [10 marks]

```
public class Recursion{
    public static void main(String[] args){
        System.out.println(method(14));
    }

    public static int method(int number){
        if (number % 7 == 3){
            return 5;
        }
        System.out.println("hello");
        return method((number % 5) + 3) - 2;
    }
}
```

The program runs main function first, it will call `method(14)`

1) method(14).

14 % 7 = 0,   0 != 3 => skip if statement

print out **"hello", then change line.**

return method((14 % 5) + 3) - 2 = method(7) - 2

2) method(7).

7 % 7 = 0,   0 != 3 => skip if statement

print out **"hello", then change line.**

return method((7 % 5) + 3) - 2 = method(5) - 2

3) method(5).

5 % 7 = 5,   5 != 3 => skip if statement

print out **"hello", then change line.**

return method((5 % 5) + 3) - 2 = method(3) - 2

4) method(3).

3 % 7 = 5,   3 == 3 =>run if statement

return 5

⇨ **method(3) = 5**

CS210-2016-January

5) Calling method(5)
   method(5) = method(3) – 2 = 5 – 2 = 3
   get 3

6) Calling method(7)
   method(7) = method(5) – 2 = 3 – 2 = 1
   get 1

7) Calling method(14)
   method(14) = method(7) – 2 = 1 – 2 = -1
   get -1

```
Therefore, the Java Program outputs
`
  hello
  hello
  hello
  -1
`
when it runs.
```

CS210-2016-January

Question b

```
public class BitManipulation{
    public static void main (String[] args){
        System.out.println(((4|6)|(5&3))<<5);
    }
}
```

The program will print out the equation
(((4|6)|(5&3))<<5)

Step 1: 4 & 6

$(4)_{10}$ $=(00000100)_2$
$(7)_{10}$ $=(00000110)_2$ |

`$(00000110)_2$ = $(6)_{10}$

Step 2: 5 & 3

$(5)_{10}$ $=(00000101)_2$
$(3)_{10}$ $=(00000011)_2$ &

`$(00000001)_2$ = $(1)_{10}$

Step 3: 6 | 1

$(6)_{10}$ $=(00000110)_2$
$(1)_{10}$ $=(00000001)_2$ &

`$(00000111)_2$ = $(7)_{10}$

Step 4: 7 << 5

$(00000111)_2$ << 5 = $(11100000)_2$ = $(224)_{10}$

Therefore, the Java Program outputs 224 when it runs.

CS210-2016-January