Question1. What will be the output of the following programs? Give reasons. [4, 7, 4]
No credit will be given if you do not give reasons (even if your output is correct). Also, if the reasoning is wrong then you will not get any credit for just writing the correct output.

```
class q1a{
  public static void sumUp(int n, int total){
    int i;
    for (i=1; i<=n; i++){
      total = total +i;
    }
  }
  public static void main(String args[]){
    int n, total;
    n=10; //n can be initialized to any +ve number
    total=0;
    sumUp(n, total);
    System.out.println(total);
  }
}
```

Answer: 0 **1 mark if correct reason is given**
Reason: Method sumUp does not return any value. The change in the local variable ***total*** is not visible outside the method. **3 marks**

```
class point{
  int p,q;
  point(int p, int q){
    this.p=p; this.q=q;
  }
  void printpoint(){
    System.out.println(this.p+" "+this.q);
  }
}

class q1b{
  public static void origin1(point p){
    p = new point(0,0);
  }
  public static void origin2(point p){
    p.p=15;
    p.q=20;
  }
  public static void main(String args[]){
    point q = new point(10,66);
    origin1(q);
    q.printpoint();
    origin2(q);
    q.printpoint();
  }
}
```

Answer:
10 66 **1 mark if correct reason given**
15 20 **1 mark if correct reason given**
Reason:
- Initially an object is created which is referred by q and has instance variables p=10 and q=66.

- When method origin1() is invoked the parameter p starts referring to the object referred by q. However, p=new point(0,0) creates a new object (instance variables p=0, q=0), and p starts referring to the new object. Since, p is a local variable, any change to p is not visible outside the method. Therefore, instance variables of q do not change
- q.printpoint(), therefore, prints values of instance variables of q which are 10 and 66 **3 marks**

- method origin2() is invoked and parameter p starts referring to object referred by q. Through use of p instance variables of object referred by q, are modified. The change is done in the object itself. Hence, instance variables of the object change to p=15 and q=20.
- q.printpoint() prints values of instance variables of q which are now 15 and 20. **2 marks**

```
class BankAccount{
  double balance;
  void deposit(double x){
    balance = balance + x;
    System.out.println("Current balance is: "+balance);
  }
}

class q1c{
  public static void main(String args[]){
    BankAccount myAccount;
    //myAccount = new BankAccount();
    myAccount.deposit(10000);
  }
}
```

Answer: The program will not compile.
1 mark if person writes program has error
2 marks if person writes compilation error
Reason: No object has been created OR myAccount has not been initialized.
2 marks

Question 2a. What will be the output of the following program? Justify your answer. [4]
No credit will be given if you do not give reasons (even if your output is correct). Also, if the reasoning is wrong then you will not get any credit for just giving the correct output.

```
class q2a{
  public static void printarray(double a[]){
    for (int i=0; i<a.length; i++)
      System.out.println(a[i]);
  }

  public static void main(String args[]){
    double x[] = {2.2, 4.4};
    double y[] = {1.1, 3.3, 5.5};
    y = x;
    x[0]=8.8;
    printarray(x);
    printarray(y);

  }
}
```

Answer:

```
8.8
4.4
8.8
4.4
```

½ mark for each output if reasoning is correct

Reason: **2 marks for the reason**

- Initially x={2.2, 4.4} and y={1.1, 3.3, 5.5}
- Assignment y=x makes y refer to the same object as being referred by x.
- Therefore, both x and y refer to array whose elements are {2.2, 4.4}
- x[0]=8.8 changes this array to {8.8, 4.4}
- printarray(x) and printarray(y) both print the elements of the same array.

Question 2b. Here is a program to copy part of an array to another array. Fill in the missing code so that the program achieves the objective. Remember that the methods have to be general to work for all correct values of the parameters. [6]

```
class q2b{
  public static void printarray(double a[]){
    for (int i=0; i<a.length; i++)
      System.out.println(a[i]);
  }

  /* copy n elements from b[] to a[]
   * store elements from a[ai] onwards
   * pick elements from b[bi] onwards
   * if a[] does not have enough slots
   * or b[] does not have enough elements
   * then no copying is done and appropriate
   * error message is flashed */

  public static void copyarray(double a[], double b[], int ai,
int bi, int n){
    if ((a.length-ai) < n)                  //1 mark
      System.out.println("Not enough slots in a[]");
    else if ((b.length-bi) < n)            //1 mark
      System.out.println("Not enough elements in b[]");
    else copyelements(a, b, ai, bi, n);      //1 mark
  }

/* copy n elements from b to a. No checks are
    required in this method  */
  public static void copyelements(double a[], double b[], int
ai, int bi, int n){
    int i, j;
    for (i=ai, j=bi; i<ai+n; i++, j++)       //2 marks
      a[i] = b[j];                            //1 mark
  }

  public static void main(String args[]){
    double x[] = {2.2, 4.4, 6.6, 8.8, 10.1};
    double y[] = {1.1, 3.3, 5.5, 7.7};
    copyarray(x,y,1,1,2); //only non –ve numbers are
                          //passed as arguments
    printarray(x);
    System.out.println();
    printarray(y);
  }
}
```

Question 3a. Here is a program written to transpose a square matrix. We do not create a new matrix; rather rows and columns are swapped in the same matrix. Complete the method transpose to achieve the goal. [5]

```
class q3a{
  public static void transpose(int a[][]){
   for (int i=0; i<a.length; i++)              //2 marks
     for (int j=i; j<a.length; j++){        //2 marks
     int t = a[i][j];                  //1 mark for swapping
     a[i][j] = a[j][i];
     a[j][i] = t;
     }
  }

  public static void main(String args[]) {
    int table[][] = {{1,2,3},{4,5,6},{7,8,9}};
    transpose(table);
   }
}
```

Only 1 mark if a new array has been created

2½ marks if the inner loop starts from 0 but everything else is correct

Question 3b. Consider the program below. This program takes a 2-D array, adds elements of each row and returns a 1-D array whose elements are sum of each row.

For example, if
table[][]={{1,2,3},{4,5},{3,5,6,9}}
then it return an array {6, 9, 23}

Complete the program below to achieve this objective. [5]

```
class q3b{
  public static double[] addUpRows(double a[][]){
     double SumA[] = new double[a.length];
     for (int i=0; i<a.length; i++)
       for (int j=0; j<a[i].length; j++)
         SumA[i] = SumA[i] + a[i][j];
     return SumA;
  }

  public static void main(String args[]) {
    double table[][] = {{1, 2, 3}, {4, 5}, {3, 5, 6, 9}};
    double SumofRows[] = addUpRows(table);
  }
}
```

//1 mark for each line

Question 4. Consider the program below that “adds one” to a single lowercase word stored in a string: starting at the rightmost character, the character is incremented (*a* becomes *b* etc.). If the character is *z* then it is transformed to *a*, and a “carry” causes the next character to the left to be incremented. If it is necessary to “carry” past the leftmost character then the string is prepended with an *a*.

For example string:
“abc” transforms to “abd”
“abz” transforms to “aca”
“zzz” transforms to “aaaa”

Briefly describe the method and complete the program below to achieve this goal. [10]

```
                                                  2 marks for description
class q4{
    public static String addone(String s){
        String t = "";
        char ch;
        int carry = 1;                            1 mark for initializations
        for (int i=s.length()-1; i>=0; i--){      1 mark for loop statement
            ch = (char)(s.charAt(i)+carry);       2 marks for update and typecast
            if (ch > 'z') {carry=1; ch='a';}      1 mark for if statement
            else carry=0;
            t = ch + t;                           1 mark for updating string t
        }
        if (carry==1) t = 'a'+ t;                 1 mark
        return t;                                 1 mark for return and type name in header
    }

//DO NOT change anything in main
  public static void main(String args[]) {
      String a = "yzzzzz";
      System.out.println(addone(a));
  }
}
```

Question 5. We have discussed bubble sort method in the class. Another important sorting method is selection sort. It makes n-1 passes through a sequence of n elements, each time moving the largest among the remaining unsorted elements into its correct position.
Given below is structure of a selection sort method. Complete the program. Trace the execution of the program by showing array after each swap. [15]

```
class q5{
   public static void sort(int a[]){
      //traverse the array repeatedly, every time
      //reducing number of elements to be sorted by one
      for (int i=a.length-1; i>0; i--){
         int m=0;
         //find index of the max element in the remaining array
         for (int j=1; j<=i; j++)
            if (a[j]>a[m]) m=j;
         //swap max and last element
         //thereby, moving the max in the last position
         swap(a,i,m);
      }
   }
   public static void swap(int a[], int i, int j){
      int t = a[i];
      a[i] = a[j];
      a[j] = t;
   }
  public static void main(String args[]) {
   int a[] = {66, 33, 99, 88, 44, 55, 22, 77};
   sort(a);
  }
}
```

Original array: 66 33 **99** 88 44 55 22 **77**
Swap 99 and 77: 66 33 77 **88** 44 55 **22** 99
Swap 88 and 22: 66 33 **77** 22 44 **55** 88 99
Swap 77 and 55: **66** 33 55 22 **44** 77 88 99
Swap 66 and 44: 44 33 **55 22** 66 77 88 99
Swap 55 and 22: **44** 33 **22** 55 66 77 88 99
Swap 44 and 22: 22 33 44 55 66 77 88 99
Nothing to be swapped: 22 33 44 55 66 77 88 99

- **NO PENALTY if output is in descending order OR student finds min element instead of max element**
- **ZERO marks if the program uses more than n swaps**
- **12 marks for the program, and 3 marks for the trace only if program is correct**
- **Deduct 7 marks if the program does not take care of duplicate numbers**
- **Deduct 4 marks if the program does not take care of negative numbers**
- **Deduct 3 marks for each mistake but if there are more than 3 mistakes then student gets 5 marks if the general conditions are correct**