Part A: Multiple choice questions (15 marks)

Read the questions and enter your answers in the answer grid at the end of this section (page 5)

1. Which one of the following assignment statements is legal?

   A) `String s = null;`
   B) `String s = 'null';`
   C) `String s = (String) 'abc';`
   D) `String s = (String) '\ufeed';`

2. Consider the following code fragment:

```
public static void main(String [] args) {
   boolean b1 = true;
   boolean b2 = false;
   boolean b3 = false;
   if ( b1 & b2 | b2 & b3 | b2 )
        System.out.print("ok ");
   if ( b1 & b2 | b2 & b3 | b2 | b1 )
        System.out.println("notOk");
    }
```

   What is printed by the code?

   A) notOk
   B) ok notOk
   C) ok
   D) Compilation error

3. What will be the output of the code below?

```
public class SwitchLoop {
   final static short x = 2;
       public static void main(String [] args) {
         for (int z=0; z < 3; z++)
         {
             switch (z){
                 case x: System.out.print("0 ");
                 case x-1: System.out.print("1 ");
                 case x-2: System.out.print("2 ");
             }
         }
     }
   }
```

   A) 0 1 2
   B) 0 1 2 1 2 2
   C) 2 1 0 1 0 0
   D) 2 1 2 0 1 2

4. What will be the output of the code below?

```
public class Test {
    public static void main(String[] args) {
        int number = 1;
        int[] numbers = new int[1];
        Test.mymethod(number, numbers);
        System.out.println("number is "+number+" and numbers[0] is " +
numbers[0]);
    }
    public static void mymethod(int x, int[] y) {
        x += 25;
        y[0] -=10;
    }
}
```

A) number is 26 and numbers[0] is -10
B) number is 1 and numbers[0] is -10
C) number is 25 and numbers[0] is 0
D) number is 10 and numbers[0] is 0

5. What is the printout of the second println statement in the main method?

```
public class Foo {
  int i;
  static int s;
  public static void main(String[] args) {
     Foo f1 = new Foo();
     System.out.println("f1.i is " + f1.i + " f1.s is " + f1.s);
     Foo f2 = new Foo();
     System.out.println("f2.i is " + f2.i + " f2.s is " + f2.s);
     Foo f3 = new Foo();
     System.out.println("f3.i is " + f3.i + " f3.s is " + f3.s);
  }
public Foo() {
  i++;
  s++;
 }
}
```

A) f2.i is 1 f2.s is 2
B) f2.i is 2 f2.s is 1
C) f2.i is 1 f2.s is 1
D) f2.i is 2 f2.s is 2

6. What is the most restrictive access modifier that will allow members of one class to have access to members of another class in the same package?

A. public
B. protected
C. default
D. private

7. What is the output of the program below?

```
public class TestArrayMethod {
public static void main(String[] args){
    int[] values = new int[5];
    passArray(values);
    for (int i = 1; i < 5; i++){
        System.out.print(values[i]+" ");
     }
  }
 public static void passArray(int[] x)  {
     x[0]=1;
     for (int i = 1; i > 5; i++){
         x[i] += x[i-1] ;
     }
 }
}
```

A) `1 2 1 2`
B) `1 2 3 4`
C) `0 0 0 0`
D) `1 0 1 0`

8. What is wrong in the following code?

```
class TempClass {
 int i;
 public void TempClass(int j) {
   int i = j;
 }
}
public class C {
  public static void main(String[] args) {
    TempClass temp = new TempClass(2);
   }
}
```

A) The program compiles fine, but it does not run because `class C` is not private
B) The program has a compilation error because `TempClass` does not have a constructor with an `int` argument
C) The program has a compilation error because `TempClass` does not have a default constructor
D) The program compiles and runs fine

9. Variables that are shared by every instances of a class are _____.

A) class variables
B) instance variables
C) public variables
D) private variables

10. What will be the output of the program below?

```
public class Question10 {
 public static void main(String[] args) {
     Count myCount = new Count(11);
     int times = 0;
     for (int i = 0; i < 100; i++) {
         increment(myCount, times);
     }
     System.out.println("myCount.count = " + myCount.count);
     System.out.println("times = " + times);
 }
 public static void increment(Count c, int times) {
     c.count++;
     times++;
 }
}
class Count {
     int count;
     Count(int c) {
         count = c;
     }
     Count() {
         count = 1;
     }
}
```

A) myCount.count = 101 times = 0
B) myCount.count = 111 times = 100
C) myCount.count = 100 times = 100
D) myCount.count = 111 times = 0

Part A: Answers

Circle your answers to the multiple choice questions for Part A in the table given.

Question	Choice			
1	A	B	C	D
2	A	B	C	D
3	A	B	C	D
4	A	B	C	D
5	A	B	C	D
6	A	B	C	D
7	A	B	C	D
8	A	B	C	D
9	A	B	C	D
10	A	B	C	D

Part B: Complete the following code fragments (15 marks)

11. Write a program that displays the first 50 prime numbers in five lines, each of which contains 10 numbers. An integer greater than 1 is prime if its only positive divisor is 1 or itself. For example, 2, 3, 5, and 7 are prime numbers, but 4, 6, 8, and 9 are not. (5 marks)

```
public class Question11 {
    public static void main(String[] args){
        int rc=0; // to count number of rows
        int count =0; // to count prime numbers
        int p=1; // start from one to ....
         int check=0; // to evaluate the given p is prime or not
       // System.out.print(1 +"\t"+2);
         while (count<=50){
             for(int i=2; i<=p/2; i++){  // to check prime
                 if(p%i==0)
                    check++;
                 }
             if(check==0){
                 System.out.print("\t"+p);
                 count++;
             }
             if(count%10==0)  // 10 prime numbers in each row
                 System.out.println();

             check=0;
             p++;

       }

    }
```

To check prime: 2 marks

To print 5 prime numbers in each row: 1.5 marks

Loop that continue until 50 prime numbers found: 1.5 marks

12. }The Calculator class below allows users to perform calculations. You are required to define the method for that class, so that the code in the TestCalculator class compiles and prints out the correct result. **You are not allowed to use any library functions / arrays /strings in order to reverse the integer**. (5 marks)

```
public class Calculator {
public void revInteger(long n){   1 mark
    while(n>0) {                  1 mark
      long y = n%10;              1 mark
      System.out.print(y);        1 mark
      n = n/10;                   1 mark
    }
  }
public class TestCalculator{
  public static void main(String[] args) {
   long n = 189;
   Calculator calc = new Calculator();
   calc.revInteger(n); // method
 }
}
```

13. The weekly hours for 8 employees are stored in a multidimensional array. Each row records an employee's seven day work hours with seven columns. The pay rate for each employee is stored in a single dimensional array. Use a nested loop to process the employee's total work hours & salary (total work hours * pay rate) from a given array. Write code to display the employee number for the employee that receives the highest salary.

(5 marks)

```
public class Question22 {
    public static void main(String[] args) {
        int[][] eWorkHours = {
            {2, 4, 3, 4, 5, 8, 8},     // employee 1
            {7, 3, 4, 3, 3, 4, 4},     // employee 2
            {3, 3, 4, 3, 3, 2, 2},     // employee 3
            {9, 3, 4, 7, 3, 4, 1},     // employee 4
            {3, 5, 4, 3, 6, 3, 8},     // employee 5
            {3, 4, 4, 6, 3, 4, 4},     // employee 6
            {3, 7, 4, 8, 3, 8, 4},     // employee 7
            {6, 3, 5, 9, 2, 7, 9}      // employee 8
        };
      double[] ePayRate={1.2,2.5,1.8,2.0,3.2,0.5,0.7,1.41}

     double HighestSalary=0;
        int eNo=0;                                  0.5 marks

        for(int i =0; i<eWorkHours.length;i++){
            int totalWorkHours=0;                    1 mark
            for (int j=0; j<eWorkHours[i].length; j++)  1 mark
            {
                totalWorkHours+=eWorkHours[i][j];  }   0.5 marks
            System.out.println(totalWorkHours);
             double salary = totalWorkHours * ePayRate[i]; 0.5 marks
             if (salary>=HighestSalary)
```

```
                    {
                        HighestSalary=salary;
                        eNo=i+1;                                    1 mark
                    }
                      }
                   System.out.printf("Employee Number:%d  Highest Salary:
%.2f" ,eNo,HighestSalary);                              0.5 marks
```

Part C Small application (30 marks)

1. Writing an AuctionItem class (15 marks)

An auction item requires the following details to be maintained in the system: `item ID, item description, seller ID, starting price and the auction status` (pending, open or closed). Once the auction has been opened the system will also need to maintain the details of the highest bidder and the amount they have bid on the item. You should start by designing and implementing a basic `AuctionItem` class, as follows:

a) Define instance variables for the `item ID`, `item description`, `seller ID`, `starting price`, `auction status` (a String which can be *"Pending", "Open" or "Closed")* and the `current highest bid`. (2 marks)

```
String itemId;
String itemName;
String sellerId;
double startingPrice;
String auctionStatus;
Double currentHighestBid;
```

b) Define a constructor for the `AuctionItem` class which accepts `the item ID, item description, seller ID and starting price` as parameters and stores the information in the corresponding instance variables. This constructor should also set the `auction status` to "*Pending*" and the `highest bid` to zero (0). (3 marks)

```
AuctionItem(String id, String name, String sId, double stPrice) {
itemId=id;
itemName= name;
startingPrice = stPrice;
auctionStatus="Pending";
currentHighestBid =0;
}
```

c) Define appropriate accessors for the each of the instance variables in this class (2 marks)

```
String getItemId() {
return itemId;
}
String getItemName() {
return itemName;
}
double getstartingPrice() {
```

```
return startingPrice;
}
String getAuctionStatus() {
return auctionStatus;
}
double  getCurrentHighestBid(){
return currentHighestBid;
}
```

d) Define a method `public boolean hasBids()`, which returns true if a valid bid has been placed on the item and false if no bid has been made (hint: a valid bid has been made if the current highest bid is equal to or greater than the starting price). (2 marks)

```
public Boolean hasBids(){
if(currentHighestBid>=startingPrice){
return true;
else
return false;
}
```

e) Define a method `public double open()`, which functions as described below: (4 marks)
If the current auction status is "*Pending*" then it changes the status to "*Open*", calculates the listing fee for the item based on the starting price and returns the result. The price structure for listing fees is as follows:

Starting Price p	**Listing Fee**
p ≤ $5.00	$0.20
$5 < p ≤ $20.00	$0.50
$20 < p ≤ $ 100.00	$1.00
$100 < p ≤ $250.00	$2.50
p > $250.00	$5.00

If the auction status is not "*Pending*" then this method should return a result of -1.

```
public double open(){
double result=0;;
double
if(auctionStatus.equals("Pending") {
 auctionStatus="Open";
 if(startingPrice<=5)
result =0.20;
else if(startingPrice<=20)
result=0.50;
else if(startingPrice<=100)
result=1.00;
```

```
else if(startingPrice<=250)
result=2.50;
else
result =5.00;
}
else
result =-1;

return result;
}
```

f) Define a method `public void print()`, which prints a summary of the details (instance variables) for the current `AuctionItem` to the screen. (2 marks)

```
public void print(){
system.out.println("Item Id : " +itemId + "    Item Description : " + itemName + "  Seller Id : "+ sellerId + "   Starting Price  : " + startingPrice + "  Auction Status  : " + auctionStatus + " Current  Highest Bid : "+ current HighestBid);
```

2. Writing an AuctionItemTest class (15 marks)

In this section you need to write an application that creates, stores, and uses a collection of `AuctionItem` objects. You will need to write code for the following in the main method:

a) Define an array of `AuctionItem` references that can store up to four objects (elements). (1 mark)

```
AuctionItem[] aucItem = new AuctionItem[4];
```

b) Create the `AuctionItem` objects specified below and store them in the array of `AuctionItem` references mentioned above. You should pass the values shown below as arguments to the constructor when creating each of the objects. (4 marks)

Item ID	Description	Seller ID	Starting Price
1	Tonka Truck	M001	1.00
2	Xbox	M002	20.00
3	Teddy Bear	M003	5.00
4	Antique Doll	M004	100.00

```
aucItem[0] = new AuctionItem("1", "Tonka Truck", "M001", 1.00);
aucItem[1] = new AuctionItem("2", "Xbox", "M002", 20.00);
aucItem[2] = new AuctionItem("3", "Teddy Bear", "M003", 5.00);
aucItem[3] = new AuctionItem("4", "Antique Doll", "M004", 100.00);
```

c) Open the auction: each of the four items that are being auctioned off by calling the open () method for each `AuctionItem` object inside a suitable loop. (3 marks)

```
public void open() {
for(int i =0 i<aucItem.length; i++) {
 if(!aucItem[i]== null)
   a[i].open();
else
break;
}
```

d) Write a static method called `findAndPrintItem()`:

- Ask the user to enter an item description.
- Find the correct item instance in the array.
- If there is no item instance with that description, display an error message: "Item not found!"
- If there is one or more item instances with that description, print the information of the items found.

```
public static void findAndPrintItem(AuctionItem[] aucItem) {
Scanner sc = new Scanner (System.in);
System.out.println("Enter the item description to search :");
String name = sc.nextLine();
int counter=0;
for (int i=0; i<aucItem.length; i++) {
    if(aucItem[i].getItemName().equals(name)) {
             aucItem[i].print();
             counter++;
        }
   }

if (counter>0)
System.out.println("The item name not found ");

}
```