Java Question and Answers

Question I 2

Solution I 3

Question II 4

Solution II 7

Question III 10

Solution III 11

Question IV 12

Solution IV 16

Question V 19

Solution V 20

Question VI 21

Solution VI 25

Question VII 29

Solution VII 30

Question VIII 31

Solution VIII 35

Java Questions with Solutions

Question I

Write main method in Solution class.

In the main method, read a String(without any numeric digits) and print the number of characters(without space) and number of spaces as in the given output.

Note: For example consider the String 'Welcome to Java', here the number of characters are 13 and number of spaces are 2. (output should be display in the same way, how it is shown in the sample output below).

Consider below sample input and output:

Input:

Welcome Home Buddy

Output:

No of characters : 16 and spaces : 2

--

Sample code snippet for reference:

Please use below code to build your solution.

--

```
public class Solution

{

        public static void main(String[] args)

        {
            //code to read values

            //code to display the result

        }



}
```

Solution I

```
import java.util.Scanner;


public class Solution {


        public static void main(String[] args) {


                Scanner sc = new Scanner(System.in);

                String inputStr = sc.nextLine();

                int count = 0,spaces=0;


                for (int i = 0; i < inputStr.length(); i++) {

                        if (inputStr.charAt(i)!= ' ') {

                                        count++;

                        }

                        if(Character.isWhitespace(inputStr.charAt(i))) {

                                spaces++;

                        }

                }

                System.out.println("No of characters : "+count+" and spaces : "+spaces);

        }
}
```

Question II

Create a Class Movie with below attributes:

movieName - String

producingComapany - String

genre - String

budget - int

Write getters, setters and parameterized constructor as required.

Create class Solution with main method.

Implement static method - budgetForGivenMovie in Solution class.

This method will take a String parameter(genre) along with the other parameter as array of Movie objects.

The method will return array of Movie objects where the String parameter appears in the genre attribute (with case sensitive search).

This method should be called from main method and if the budget is greater than 80000000 print "High Budget Movie" else print "Low budget movie".

Before calling this method(budgetForGivenMovie) in the main method, use Scanner object to read values for four Movie objects referring the attributes in the above sequence.

then, read the value for search parameter.

Next call the method budgetForGivenMovie and display the result.

Consider below sample input and output:

Input:

aaa

DC

Action

250000000

bbb

Marvel

Action

100000000

ccc

Paramount

Drama

75000000

ddd

Fox

Romance

20000000

Action

Output:

High Budget Movie

High Budget Movie

--

Sample code snippet for reference:

Please use below code to build your solution.

--

public class Solution

```
{

 public static void main(String[] args)
 {
  //code to read values
  //code to call required method
  //code to display the results
 }

 public static Movie[] budgetForGivenMovie(Movie[] t, String string)
 {
  //method logic


 }
}

class Movie
{
  //code to build Associate class
}
```

Note on using Scanner object:

Sometimes scanner does not read the new line character while invoking methods like nextInt(), nextDouble() etc.

Usually, this is not an issue, but this may be visible while calling nextLine() immediately after those methods.

Consider below input values:

22

hello

Referring below code:

```
Scanner sc = new Scanner(System.in);

int x = sc.nextInt();

String str = sc.nextLine(); -> here we expect str to have value hello. Instead it may be "".
```

If above issue is observed, then it is suggested to add one more explicit call to nextLine() after reading numeric value.

Solution II

```
import java.util.Scanner;


public class Solution {

        public static void main(String[] args) {

                Movie[] movie = new Movie[4];

                String search;


    Scanner sc = new Scanner(System.in);


    for(int i = 0;i< movie.length;i++)

    {

      String movieName = sc.nextLine();

      String producingCompany = sc.nextLine();

      String genre = sc.nextLine();

      int budget = sc.nextInt();sc.nextLine();
```

```
        movie[i] = new Movie(movieName, producingCompany, genre, budget);
      }
      search = sc.nextLine();
      Movie[] newMovie = budgetForGivenMovie(movie,search);
      for(int i =0;i<newMovie.length;i++){
                  if( newMovie[i].getBudget()>80000000){
                  System.out.println("High Budget Movie");
                  }else{
                        System.out.println("Low budget movie");
                  }
         }
}

         private static Movie[] budgetForGivenMovie(Movie[] t, String genre) {
               int c=0;
               for(int i=0;i<t.length;i++){
                     if(t[i].getGenre().equals(genre)){
                           c++;
                     }
               }
               int b=0;
               Movie[] m=new Movie[c];
               for(int i=0;i<t.length;i++){
                     if(t[i].getGenre().equals(genre)){
                           m[b]=t[i];
                           b++;
                     }


               }
```

```
			return m;
		}
}
class Movie {
		private String movieName;
		private String producingComapany;
		private String genre;
		private int budget;
		public String getMovieName() {
			return movieName;
		}
		public void setMovieName(String movieName) {
			this.movieName = movieName;
		}
		public String getProducingComapany() {
			return producingComapany;
		}
		public void setProducingComapany(String producingComapany) {
			this.producingComapany = producingComapany;
		}
		public String getGenre() {
			return genre;
		}
		public void setGenre(String genre) {
			this.genre = genre;
		}
		public int getBudget() {
			return budget;
		}
```

```
	public void setBudget(int budget) {
		this.budget = budget;
	}
	public Movie(String movieName, String producingComapany, String genre, int budget) {
		super();
		this.movieName = movieName;
		this.producingComapany = producingComapany;
		this.genre = genre;
		this.budget = budget;
	}

}
```

Question III

Write main method in Solution class.

In the main method, read a String(without any numeric digits) and print the last characters of each word in the given String.

Note: For example consider the String 'Welcome to Java', here the last characters of each word "eoa" should be displayed in the same sequence as it appears in the given input.

Consider below sample input and output:

Input:

Welcome Home BuddY

Output:

eeY

Sample code snippet for reference:

Please use below code to build your solution.

```
public class Solution
{

        public static void main(String[] args)
        {
                //code to read values
                //code to display the result
        }


}
```

Solution III

```
import java.util.Scanner;

public class Solution {


        static void printLastCharacter(String str)
  {

     str = str + " ";
```

```
      for (int i = 0; i < str.length(); i++) {

        if (str.charAt(i) == ' ')

          System.out.print(str.charAt(i - 1));
      }
    }



        public static void main(String[] args) {
                Scanner sc = new Scanner(System.in);
                String inputStr = sc.nextLine();
                printLastCharacter(inputStr);
        }

}
```

Question IV

Create a Class Inventory with below attributes:

inventoryId - String

maximumQuantity - int

currentQuantity - int

threshold - int

Write getters, setters and parameterized constructor as required.

Create class Solution with main method.

Implement static method - replenish in Solution class.

This method will take an int parameter named limit along with the other parameter as array of Inventory objects.

The method will return array of Inventory where the threshold attribute is less than or equal to the int parameter passed.

This method should be called from main method and display the id of returned objects along with Filling status.

if the threshold is greater than or equal to 75 then it should print "Critical Filling" as Filling Status. If the threshold is between 74 to 50 then Filling status should be "Moderate Filling", else should be "Non-Critical Filling" .

Before calling this method(replenish) in the main method, use Scanner object to read values for four Inventory objects referring the attributes in the above sequence.

then, read the value for limit parameter.

Next call the method replenish and display the result.

Consider below sample input and output:

Input:

```
1
100
50
40
2
100
50
50
3
100
```

40

45

4

100

80

25

45

Output:

1 Non-Critical Filling

3 Non-Critical Filling

4 Non-Critical Filling

Sample code snippet for reference:

Please use below code to build your solution.

```
public class Solution
{

 public static void main(String[] args)
 {
  //code to read values
  //code to call required method
  //code to display the results
 }
 public static Inventory[] replenish(int limit, Inventory[] inventory)
 {
  //method logic


 }
}


class Inventory
{
  //code to build Inventory class
}
```

Note on using Scanner object:

Sometimes scanner does not read the new line character while invoking methods like nextInt(), nextDouble() etc.

Usually, this is not an issue, but this may be visible while calling nextLine() immediately after those methods.

Consider below input values:

```
22
hello
```

Referring below code:

```
Scanner sc = new Scanner(System.in);
int x = sc.nextInt();
```

String str = sc.nextLine(); -> here we expect str to have value hello. Instead it may be "".

If above issue is observed, then it is suggested to add one more explicit call to nextLine() after reading numeric value.

Solution IV

```
import java.util.Scanner;

public class Solution {
        public static void main(String[] args) {
                Inventory[] inventory = new Inventory[4];

                Scanner sc = new Scanner(System.in);

                for (int i = 0; i < inventory.length; i++) {
                        int inventoryId = sc.nextInt();
                        sc.nextLine();
                        int maximumQuantity = sc.nextInt();
                        int currentQuantity = sc.nextInt();
                        int threshold = sc.nextInt();
                        sc.nextLine();
                        inventory[i] = new Inventory(inventoryId, maximumQuantity,
currentQuantity, threshold);
                }
                int limit = sc.nextInt();
                sc.close();
                Inventory[] newInventory = replenish(inventory,limit);
                for (int i = 0; i < newInventory.length; i++) {
                        if (newInventory[i].getThreshold() >= 75) {
```

```
                                        System.out.println(newInventory[i].getInventoryId()+" Critical
Filling");
                              } else if (newInventory[i].getThreshold() <= 74 &&
newInventory[i].getThreshold() >= 50) {
                                        System.out.println(newInventory[i].getInventoryId()+ " Moderate
Filling");
                              } else {
                                        System.out.println(newInventory[i].getInventoryId()+" Non-Critical
Filling");
                              }
                    }
          }

          public static Inventory[] replenish(Inventory[] inventory,int limit) {
                    int count = 0;
                    for (int i = 0; i < inventory.length; i++) {
                              if (inventory[i].getThreshold() <= limit) {
                                        count++;
                              }
                    }
                    int index = 0;
                    Inventory[] newInventory = new Inventory[count];
                    for (int i = 0; i < inventory.length; i++) {
                              if (inventory[i].getThreshold() <= limit) {
                                        newInventory[index] = inventory[i];
                                        index++;
                              }
                    }
                    return newInventory;
          }
```

```
}

class Inventory {
	private int inventoryId;
	private int maximumQuantity;
	private int currentQuantity;
	private int threshold;

	public int getInventoryId() {
		return inventoryId;
	}

	public void setInventoryId(int inventoryId) {
		this.inventoryId = inventoryId;
	}

	public int getMaximumQuantity() {
		return maximumQuantity;
	}

	public void setMaximumQuantity(int maximumQuantity) {
		this.maximumQuantity = maximumQuantity;
	}

	public int getCurrentQuantity() {
		return currentQuantity;
	}

	public void setCurrentQuantity(int currentQuantity) {
		this.currentQuantity = currentQuantity;
	}

	public int getThreshold() {
		return threshold;
	}

	public void setThreshold(int threshold) {
		this.threshold = threshold;
	}

	public Inventory(int inventoryId, int maximumQuantity, int currentQuantity, int threshold) {
		super();
		this.inventoryId = inventoryId;
		this.maximumQuantity = maximumQuantity;
		this.currentQuantity = currentQuantity;
		this.threshold = threshold;
	}
}
```

Question V

Write main method in Solution class.

In the main method, read a String (which may have alphabets along with numeric digits) and print the number of vowel and consonants (any alphabet apart from vowel is a consonant) present in the given String.

Note: The output should be printed in the same format as mentioned in the sample output.

Consider below sample input and output:

Input:

Welcome123

Output:

Number of Vowels: 3

Number of Consonants: 4

Sample code snippet for reference:

Please use below code to build your solution.

```
public class Solution

{


        public static void main(String[] args)

        {

                //code to read values

                //code to display the result

        }



}
```

Solution V

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

```
public class Solution {

        public static void main(String[] args) {
                Scanner sc = new Scanner(System.in);
                String inputStr = sc.nextLine();
                int count = 0;
                int vowCount=0;
                int conCount=0;
                for(int i=0;i<inputStr.length();i++) {
                        char ch = inputStr.charAt(i);
                        if(ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
                                vowCount++;
                                }
                        else if((ch >= 'a'&& ch <= 'z' || ch >= 'A'&& ch <= 'Z')) {
                                conCount++;
      }
    }
    System.out.println("Number of Vowels: " + vowCount);
    System.out.println("Number of Consonants: " + conCount);
                }
        }
```

Question VI

Create a Class Player with below attributes:

playerId - int

playerName - String

iccRank - int

noOfMatchesPlayed - int

totalRunsScored - int

Write getters, setters and parameterized constructor as required.

Create class Solution with main method.

Implement static method - findAverageScoreOfPlayers in Solution class.

This method will take an int parameter named target along with the other parameter as array of Player objects.

The method will calculate the average runrate of the player based on totalRunsScored and noOfMatchesPlayed and return the same in a double array where the noOfMatchesPlayed attribute is greater than or equal to the int parameter target passed.

This method should be called from the main method and display the Grade of the players.

If the averageRunRate is greater than or equal to 80 then it should print "Grade A Player". If the averageRunRate is between 50 to 79 then it should print "Grade B Player", else it should print "Grade C Player" .

Before calling this method(findAverageScoreOfPlayers) in the main method, use Scanner object to read values for four Player objects referring the attributes in the above sequence.

then, read the value for noOfMatchesPlayed parameter.

Next call the method findAverageScoreOfPlayers, write the logic to display the Grade(in the main method) and display the result.

Consider below sample input and output:

Input:

100

Sachin

```
1
102
13000
101
Shewag
2
110
10000
102
Dhoni
3
80
7500
104
Kholi
4
70
7000
100
```

Output:

```
Grade A Player
Grade A Player
```

--

Sample code snippet for reference:

Please use below code to build your solution.

```
public class Solution

{


 public static void main(String[] args)

 {

  //code to read values

  //code to call required method

  //code to calculate Grade and display the results

 }


 public static double[] findAverageScoreOfPlayers(Player[] player,int totalMatches)

 {

  //method logic


 }

}


class Player

{

  //code to build player class

}
```

Note on using Scanner object:

Sometimes scanner does not read the new line character while invoking methods like nextInt(), nextDouble() etc.

Usually, this is not an issue, but this may be visible while calling nextLine() immediately after those methods.

Consider below input values:

22

hello

Referring below code:

```
Scanner sc = new Scanner(System.in);

int x = sc.nextInt();

String str = sc.nextLine(); -> here we expect str to have value hello. Instead it may be "".
```

If above issue is observed, then it is suggested to add one more explicit call to nextLine() after reading numeric value.

Solution VI

```
import java.util.Scanner;

public class Solution {
        public static void main(String[] args) {
        Player[] players = new Player[4];

        Scanner sc = new Scanner(System.in);
        for(int i=0;i<players.length;i++) {
                int playerId = sc.nextInt();
                sc.nextLine();
                String playerName = sc.nextLine();
                int iccRank = sc.nextInt();
                int noOfMatchesPlayed = sc.nextInt();
                int noOfRunsScored = sc.nextInt();
                sc.nextLine();
                players[i] = new
Player(playerId,playerName,iccRank,noOfMatchesPlayed,noOfRunsScored);
            }
            int target = sc.nextInt();
            double[] avgRunRate = findAverageScoreOfPlayers(players,target);

            for(int i=0;i<avgRunRate.length;i++) {
                if(avgRunRate[i]>=80.0) {
                    System.out.println("Grade A Player");
                }else if(avgRunRate[i]<=79.0&&avgRunRate[i]>=50.0){
                    System.out.println("Grade B Player");
                }else {
                    System.out.println("Grade C Player");
                }
            }
}
            public static double[] findAverageScoreOfPlayers(Player[] players,int target) {
                int count=0;
                for(int i=0;i<players.length;i++) {
                    if(players[i].getNoOfMatchesPlayed()>=target) {
                        count++;
                    }
                }

                double averageRunRate = 0.0;
                double[] avgRunRateArray = new double[count];
                for(int i=0;i<players.length;i++) {
                        if(players[i].getNoOfMatchesPlayed()>=target) {
                                averageRunRate =
Double.valueOf(players[i].getTotalRunsScored()/players[i].getNoOfMatchesPlayed());
                                avgRunRateArray[i] = averageRunRate;
                        }
                }
                return avgRunRateArray;
        }
}
class Player{
        private int playerId;
        private String playerName;
        private int iccRank;
        private int noOfMatchesPlayed;
        private int totalRunsScored;
        public int getPlayerId() {
                return playerId;
        }
        public void setPlayerId(int playerId) {
                this.playerId = playerId;
        }
        public String getPlayerName() {
                return playerName;
        }
        public void setPlayerName(String playerName) {
                this.playerName = playerName;
        }
```

```
	public int getIccRank() {
		return iccRank;
	}
	public void setIccRank(int iccRank) {
		this.iccRank = iccRank;
	}
	public int getNoOfMatchesPlayed() {
		return noOfMatchesPlayed;
	}
	public void setNoOfMatchesPlayed(int noOfMatchesPlayed) {
		this.noOfMatchesPlayed = noOfMatchesPlayed;
	}
	public int getTotalRunsScored() {
		return totalRunsScored;
	}
	public void setTotalRunsScored(int totalRunsScored) {
		this.totalRunsScored = totalRunsScored;
	}
	public Player(int playerId, String playerName, int iccRank, int noOfMatchesPlayed, int
totalRunsScored) {
		super();
		this.playerId = playerId;
		this.playerName = playerName;
		this.iccRank = iccRank;
		this.noOfMatchesPlayed = noOfMatchesPlayed;
		this.totalRunsScored = totalRunsScored;
	}
```

}

Question VII

Write main method in Solution class.

In the main method, write code to read a String value using Scanner and print the smallest vowel. Assume all input values are in lower case.

E.g If the input value is "matrix" then output will be a (since there are two vowels a and i where a is smaller as per ASCII sequence).

Sample code snippet for reference:

Please use below code to build your solution.

```
public class Solution

{

          public static void main(String[] args)

          {

                    //code to read value

                    //code to display the result

          }

}
```

Solution VII

```
import java.util.Scanner;


public class Solution {


        public static void main(String[] args)

        {

                Scanner sc = new Scanner(System.in);


                String str = sc.nextLine();


                char t = 'u';


                for(int i = 0;i<str.length();i++)

                {

                        char x = str.charAt(i);


                        if(x=='a'||x=='e'||x=='i'||x=='o'||x=='u')

                        {

                                if(t>str.charAt(i))

                                        t = str.charAt(i);

                        }

                }

                System.out.println(t);

        }

}
```

Question VIII

Create a class Sim with below attributes:

simId - int

customerName - String

balance - double

ratePerSecond - double

circle - String

Write getters, setters and parameterized constructor as required.

Public class Solution is already created with main method.

Code inside main method should not be altered else your solution might be scored as zero.

You may copy the code from main method in eclipse to verify your implementation.

Implement static method - transferCustomerCircle in Solution class.

This method will take first parameter as array of Sim class objects, second parameter as circle to be transferred (which is String parameter circle1) and third parameter as new circle (which is String parameter circle2).

Method will transfer the customer to new circle (circle2), where the circle attribute would match second parameter (circle1). Method will return array of Sim objects for which circle is transferred. Return array should be sorted in descending order of ratePerSecond (assuming ratePerSecond is not same for any of the Sim objects).

This method should be called from main method and display the simId,customerName,circle and ratePerSecond of returned objects (as per sample output).

Main method mentioned above already has Scanner code to read values, create objects and test above methods. Hence do not modify it.

Consider below sample input and output:

Input:

1

raj

100

1.5

KOL

2

chetan

200

1.6

AHD

3

asha

150

1.7

MUM

4

kiran

50

2.2

AHD

5

vijay

130

```
1.8
AHD
AHD
KOL
```

Output:

```
4 kiran KOL 2.2
5 vijay KOL 1.8
2 chetan KOL 1.6
```

Sample code snippet for reference:

Please use below code to build your solution. Make sure that Solution class is public and Sim class is not public.

```
import java.util.Scanner;


public class Solution
{

 public static void main(String[] args)
 {
                        Sim[] cards = new Sim[5];

                        Scanner sc = new Scanner(System.in);

                        for(int i = 0;i<5;i++)
                        {
                                    int simId = sc.nextInt();sc.nextLine();
                        String customerName = sc.nextLine();
                        double balance = sc.nextDouble();
                        double ratePerSecond = sc.nextDouble();sc.nextLine();
                        String circle = sc.nextLine();

                        cards[i] = new Sim(simId,customerName, balance,ratePerSecond, circle);
                }

                String circle1 = sc.nextLine();
                String circle2 = sc.nextLine();

                Sim[] result = transferCustomerCircle(cards, circle1, circle2);

                for(Sim s: result)
                        System.out.println(s.getSimId()+" "+s.getCustomerName()+" "
                +s.getCircle() + " " + s.getRatePerSecond());

}

public static Sim[] transferCustomerCircle(Sim[] cards, String circle1, String circle2)
{
 //method logic

}
}

class Sim
{
 //code to build Sim class
```

```
}
```

Solution VIII

```
import java.util.Scanner;

public class Solution {

	public static void main(String[] args) {
		// TODO Auto-generated method stub

		Sim[] cards = new Sim[5];

		Scanner sc = new Scanner(System.in);

		for(int i = 0;i<5;i++)
		{
			int simId = sc.nextInt();sc.nextLine();
			String customerName = sc.nextLine();
			double balance = sc.nextDouble();
			double ratePerSecond = sc.nextDouble();sc.nextLine();
			String circle = sc.nextLine();

			cards[i] = new Sim(simId,customerName, balance,ratePerSecond, circle);
		}
```

```
        String circle1 = sc.nextLine();
        String circle2 = sc.nextLine();

        Sim[] result = transferCustomerCircle(cards, circle1, circle2);

        for(Sim s: result)
                System.out.println(s.getSimId()+" "+s.getCustomerName()+" "
        +s.getCircle() + " " + s.getRatePerSecond());

}

public static Sim[] transferCustomerCircle(Sim[] cards, String circle1, String circle2)
{
        int count = 0;

        for(Sim s: cards)
        {
                if(s.getCircle().equals(circle1))
                        count++;
        }

        Sim[] result = new Sim[count];

        count = 0;

        for(Sim s: cards)
        {
                if(s.getCircle().equals(circle1))
```

```
                    {
                              s.setCircle(circle2);
                              result[count++] = s;
                    }
          }

          for(int i = 0;i<result.length;i++)
          {
                    for(int j = 0;j<i;j++)
                    {
                              if(result[i].getRatePerSecond()>
                              result[j].getRatePerSecond())
                              {
                                        Sim temp = result[i];
                                        result[i] = result[j];
                                        result[j] = temp;
                              }
                    }
          }

          return result;
     }
}

class Sim
{
          int simId;
          String customerName;
          double balance;
```

```
        double ratePerSecond;

        String circle;

        public Sim(int simId, String customerName, double balance, double ratePerSecond, String
circle) {

                this.simId = simId;

                this.customerName = customerName;

                this.balance = balance;

                this.ratePerSecond = ratePerSecond;

                this.circle = circle;

        }

        public int getSimId() {

                return simId;

        }

        public void setSimId(int simId) {

                this.simId = simId;

        }

        public String getCustomerName() {

                return customerName;

        }

        public void setCustomerName(String customerName) {

                this.customerName = customerName;

        }

        public double getBalance() {

                return balance;

        }

        public void setBalance(double balance) {

                this.balance = balance;

        }

        public double getRatePerSecond() {
            return ratePerSecond;
        }
        public void setRatePerSecond(double ratePerSecond) {
            this.ratePerSecond = ratePerSecond;
        }
        public String getCircle() {
            return circle;
        }
        public void setCircle(String circle) {
            this.circle = circle;
        }


    }
```