PRACTICAL-1

- **Preprocessing of missing values. Replace the missing values for given automobile dataset "imports-85.data" with user specified global constant.**

Program :

```
package a;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;

public class DM_Practical_1 {

	FileReader fin;
	FileWriter fout;
	BufferedReader bin;
	BufferedWriter bout;
	String[] cols;
	ArrayList<String[]> dataList;
	String initalData[];
	String line = null;
	// ArrayList<String> col;

	public DM_Practical_1() throws IOException {
		intialData();
		fileRead();
		fillMissing();
		fileWrite();
	}
	private void fileRead() throws IOException {
		int i = 0;
		dataList = new ArrayList<>();
		fin = new FileReader(new File("assets/imports-85.data"));
		bin = new BufferedReader(fin);

		while ((line = bin.readLine()) != null) {
			cols = line.split(",");
			dataList.add(cols);

		}
		// TODO Auto-generated method stub

	}

	private void fileWrite() throws IOException {
		fout=new FileWriter(new File("assets/imports-85-1.data"));
		bout=new BufferedWriter(fout);
		for(String[]rows : dataList)
		{
```

```
                    line=String.join(",",rows);
                    bout.write(line+"\n");
                }
                bout.close();
                bin.close();
                fin.close();
                fout.close();
                // TODO Auto-generated method stub

        }

        private void fillMissing() {
                int j = 0;
                for (String[] rows : dataList) {
                        int i = 0;
                        for (String cols : rows) {
                                //System.out.print(i + "\t" + cols);
                                if (cols.contains("?"))
                                        dataList.get(j)[i] = initalData[i];
                                i++;
                        }
                        //System.out.println();
                        j++;
                }

        }

        private void intialData() {
                initalData = new String[26];
                initalData[1] = 46 + "";
                initalData[5] = "three";
                initalData[18] = 4.6f + "";
                initalData[19] = 4.6f + "";
                initalData[21] = 46 + "";
                initalData[22] = 46000 + "";
                initalData[25] = 46000 + "";
        }
        public static void main(String arg[]) throws IOException {
                new DM_Practical_1();
        }
}
```

Output :-

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	symboling	normalize	make	fuel-type	aspiration	num-of-d	body-styl	drive-whe	engine-lo	wheel-ba	length	width	height	curb-weig	engine-ty	num-of-c	engine-si	fuel-syste	bore	stroke
2	3	46	alfa-rome	gas	std	two	convertib	rwd	front	88.6	168.8	64.1	48.8	2548	dohc	four	130	mpfi	3.47	2.68
3	3	46	alfa-rome	gas	std	two	convertib	rwd	front	88.6	168.8	64.1	48.8	2548	dohc	four	130	mpfi	3.47	2.68
4	1	46	alfa-rome	gas	std	two	hatchback	rwd	front	94.5	171.2	65.5	52.4	2823	ohcv	six	152	mpfi	2.68	3.47
5	2	164	audi	gas	std	four	sedan	fwd	front	99.8	176.6	66.2	54.3	2337	ohc	four	109	mpfi	3.19	3.4
6	2	164	audi	gas	std	four	sedan	4wd	front	99.4	176.6	66.4	54.3	2824	ohc	five	136	mpfi	3.19	3.4
7	2	46	audi	gas	std	two	sedan	fwd	front	99.8	177.3	66.3	53.1	2507	ohc	five	136	mpfi	3.19	3.4
8	1	158	audi	gas	std	four	sedan	fwd	front	105.8	192.7	71.4	55.7	2844	ohc	five	136	mpfi	3.19	3.4
9	1	46	audi	gas	std	four	wagon	fwd	front	105.8	192.7	71.4	55.7	2954	ohc	five	136	mpfi	3.19	3.4
10	1	158	audi	gas	turbo	four	sedan	fwd	front	105.8	192.7	71.4	55.9	3086	ohc	five	131	mpfi	3.13	3.4
11	0	46	audi	gas	turbo	two	hatchback	4wd	front	99.5	178.2	67.9	52	3053	ohc	five	131	mpfi	3.13	3.4
12	2	192	bmw	gas	std	two	sedan	rwd	front	101.2	176.8	64.8	54.3	2395	ohc	four	108	mpfi	3.5	2.8
13	0	192	bmw	gas	std	four	sedan	rwd	front	101.2	176.8	64.8	54.3	2395	ohc	four	108	mpfi	3.5	2.8
14	0	188	bmw	gas	std	two	sedan	rwd	front	101.2	176.8	64.8	54.3	2710	ohc	six	164	mpfi	3.31	3.19
15	0	188	bmw	gas	std	four	sedan	rwd	front	101.2	176.8	64.8	54.3	2765	ohc	six	164	mpfi	3.31	3.19
16	1	46	bmw	gas	std	four	sedan	rwd	front	103.5	189	66.9	55.7	3055	ohc	six	164	mpfi	3.31	3.19
17	0	46	bmw	gas	std	four	sedan	rwd	front	103.5	189	66.9	55.7	3230	ohc	six	209	mpfi	3.62	3.39
18	0	46	bmw	gas	std	two	sedan	rwd	front	103.5	193.8	67.9	53.7	3380	ohc	six	209	mpfi	3.62	3.39
19	0	46	bmw	gas	std	four	sedan	rwd	front	110	197	70.9	56.3	3505	ohc	six	209	mpfi	3.62	3.39
20	2	121	chevrolet	gas	std	two	hatchback	fwd	front	88.4	141.1	60.3	53.2	1488	l	three	61	2bbl	2.91	3.03

PRACTICAL-2

- **Preprocessing of missing values. Replace the missing values for given automobile dataset "imports-85.data" with mean, median and mode value of numeric attribute.**

  Program :-

```
package a;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Enumeration;
import java.util.Hashtable;
import java.util.Scanner;
import java.util.Set;


public class DM_Practical_2 {

	int choice = 0;
	FileReader fin;
	FileWriter fout;
	BufferedReader bin;
	BufferedWriter bout;
	String[] cols;
	ArrayList<String[]> dataList;
	Double initalData[][];
	String line = null;
	int missingCol[];

	public DM_Practical_2() throws IOException {
		initalData = new Double[26][3];
		fileRead();
		Scanner s = new Scanner(System.in);
		System.out.println("Menu\n1 : Mean\n2 : Meaden\n3 : Mode\n0 : Exit\n");
		choice = s.nextInt();
		while (true) {
			if (choice == 0)
				break;
			fillMissing();
			printIntialData();
			fillMissingDatawithMean();
			// printData();
			fileWrite();
			choice = s.nextInt();
		}
	}

	public static void main(String[] args) throws IOException {
		// TODO Auto-generated method stub
		new DM_Practical_2();
	}

	private void printIntialData() {
		for (Double a[] : initalData)
```

```
                System.out.println("Sum : " + a[0] + "\t NO :" + a[1] + " Mean : " + a[2]);
        }

        private void printData() {
                for (String[] a : dataList) {
                        for (String cols : a) {
                                System.out.println(cols);
                        }
                }
        }

        private void fileRead() throws IOException {
                int i = 0;
                dataList = new ArrayList<>();
                fin = new FileReader(new File("assets/imports-85.data"));
                bin = new BufferedReader(fin);

                while ((line = bin.readLine()) != null) {
                        cols = line.split(",");
                        dataList.add(cols);

                }
        }

        private void fileWrite() throws IOException {
                fout = new FileWriter(new File("assets/imports-85-2.data"));
                bout = new BufferedWriter(fout);
                for (String[] rows : dataList) {
                        line = String.join(",", rows);
                        bout.write(line + "\n");
                }
                bout.close();
                bin.close();
                fin.close();
                fout.close();
                // TODO Auto-generated method stub

        }

        private void fillMissing() {
                missingCol = new int[26];
                int j = 0;
                for (String[] rows : dataList) {
                        int i = 0;
                        for (String cols : rows) {
                                // System.out.print(i + "\t" + cols);
                                if (cols.contains("?"))
                                        missingCol[i] = 1;
                                // dataList.get(j)[i] = initalData[i];
                                i++;
                        }
                        // System.out.println();
                        j++;
                }
                for (int i = 0; i < missingCol.length; i++) {
                        if (missingCol[i] == 1) {
                                calculateMissingData(i);
                        }
                }
        }

        private void calculateMissingData(int i) {
```

```
		String mode[][] = null;
		//if (choice == 1 || choice == 2)
			if (i == 5)
				return;

		int j = 0;
		System.out.println("cols : " + i);
		double sum = 0, n = 0;
		for (String[] rows : dataList) {
			if (!rows[i].contains("?"))
				try {
					if (choice == 1)
						sum = sum + Double.parseDouble(rows[i]);
					n++;
					j++;
					} catch (Exception e) {
					e.printStackTrace();
				}
		}
		intialData(i, sum, (double) n);
	}
	private void intialData(int index, double sum, double cnt) {
		if (choice == 1) {
			initalData[index][0] = sum;
			initalData[index][1] = cnt;
			initalData[index][2] = sum / cnt;
		}
	}
	private void fillMissingDatawithMean() {
		int j = 0;
		for (String[] rows : dataList) {
			int i = 0;
			for (String cols : rows) {
				// System.out.print(i + "\t" + cols);
				if (cols.contains("?"))
					if (initalData[i][2] != null)
						dataList.get(j)[i] = String.valueOf(initalData[i][2]);
				i++;
			}
			// System.out.println();
			j++;
		}
	}
}
```

Output :-

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	symboling	normalize	make	fuel-type	aspiration	num-of-d	body-styl	drive-whe	engine-lo	wheel-ba	length	width	height	curb-weig	engine-ty	num-of-cy	engine-si	fuel-syste	bore	stroke
2	3	122	alfa-rome	gas	std	two	convertibl	rwd	front	88.6	168.8	64.1	48.8	2548	dohc	four	130	mpfi	3.47	2.68
3	3	122	alfa-rome	gas	std	two	convertibl	rwd	front	88.6	168.8	64.1	48.8	2548	dohc	four	130	mpfi	3.47	2.68
4	1	122	alfa-rome	gas	std	two	hatchback	rwd	front	94.5	171.2	65.5	52.4	2823	ohcv	six	152	mpfi	2.68	3.47
5	2	164	audi	gas	std	four	sedan	fwd	front	99.8	176.6	66.2	54.3	2337	ohc	four	109	mpfi	3.19	3.4
6	2	164	audi	gas	std	four	sedan	4wd	front	99.4	176.6	66.4	54.3	2824	ohc	five	136	mpfi	3.19	3.4
7	2	122	audi	gas	std	two	sedan	fwd	front	99.8	177.3	66.3	53.1	2507	ohc	five	136	mpfi	3.19	3.4
8	1	158	audi	gas	std	four	sedan	fwd	front	105.8	192.7	71.4	55.7	2844	ohc	five	136	mpfi	3.19	3.4
9	1	122	audi	gas	std	four	wagon	fwd	front	105.8	192.7	71.4	55.7	2954	ohc	five	136	mpfi	3.19	3.4
10	1	158	audi	gas	turbo	four	sedan	fwd	front	105.8	192.7	71.4	55.9	3086	ohc	five	131	mpfi	3.13	3.4
11	0	122	audi	gas	turbo	two	hatchback	4wd	front	99.5	178.2	67.9	52	3053	ohc	five	131	mpfi	3.13	3.4
12	2	192	bmw	gas	std	two	sedan	rwd	front	101.2	176.8	64.8	54.3	2395	ohc	four	108	mpfi	3.5	2.8
13	0	192	bmw	gas	std	four	sedan	rwd	front	101.2	176.8	64.8	54.3	2395	ohc	four	108	mpfi	3.5	2.8
14	0	188	bmw	gas	std	two	sedan	rwd	front	101.2	176.8	64.8	54.3	2710	ohc	six	164	mpfi	3.31	3.19
15	0	188	bmw	gas	std	four	sedan	rwd	front	101.2	176.8	64.8	54.3	2765	ohc	six	164	mpfi	3.31	3.19
16	1	122	bmw	gas	std	four	sedan	rwd	front	103.5	189	66.9	55.7	3055	ohc	six	164	mpfi	3.31	3.19
17	0	122	bmw	gas	std	four	sedan	rwd	front	103.5	189	66.9	55.7	3230	ohc	six	209	mpfi	3.62	3.39
18	0	122	bmw	gas	std	two	sedan	rwd	front	103.5	193.8	67.9	53.7	3380	ohc	six	209	mpfi	3.62	3.39
19	0	122	bmw	gas	std	four	sedan	rwd	front	110	197	70.9	56.3	3505	ohc	six	209	mpfi	3.62	3.39
20	2	121	chevrolet	gas	std	two	hatchback	fwd	front	88.4	141.1	60.3	53.2	1488	l	three	61	2bbl	2.91	3.03

PRACTICAL-3

- **Preprocessing of missing values. Replace the missing values for given automobile dataset "imports-85.data" with mean value of each attribute class. (Consider no. of doors as the class attribute - 6th attribute)**

Program :-

```
import java.io.*;
public class Practical_3 {
	public static void main(String[] atr) throws Exception
	{
		FileReader fr=new FileReader("imports-85.csv");
		BufferedReader br= new BufferedReader(fr);
		FileWriter fw=new FileWriter("output.csv");
		String line =br.readLine();
		line=br.readLine();
		int total1=0,total2=0,total3=0,b[],i,j=0,k=0,l=0;
		b=new int[164];
		while(line!=null)
		{
			String[] arr=line.split(",");
			//When no.of doors are four
			if(arr[5].equals("four"))
				{
					if(!arr[1].equalsIgnoreCase("?"))
					{
						total1=total1+Integer.parseInt(arr[1]);
						b[j]=Integer.parseInt(arr[1]);
						j++;
					}}
				if(arr[5].equals("two"))
				{
					if(!arr[1].equalsIgnoreCase("?"))
					{
					total2=total2+Integer.parseInt(arr[1]);
					b[k]=Integer.parseInt(arr[1]);
					k++;
					}
				}
				if(arr[5].equals("?"))
				{
					if(!arr[1].equalsIgnoreCase("?"))
					{
					total3=total3+Integer.parseInt(arr[1]);
					b[l]=Integer.parseInt(arr[1]);
					l++;
					}}
				line=br.readLine();
		}
		double mean1=0.0,mean2=0.0,mean3=0.0;

		mean1=(double)total1/j; //When no.of doors are four
		mean2=(double)total2/k; //When no.of doors are two
```

```
mean3=(double)total3/l; //When no.of doors are ?
System.out.println("Mean1 is: "+ mean1+"\nMean2 is: "+ mean2+"\nMean3 is: "+ mean3+"\n");
FileReader fr1=new FileReader("imports-85.csv");
BufferedReader br1= new BufferedReader(fr1);
String  line1=br1.readLine();
line1=br1.readLine();
while(line1!=null)
{
        String arr[]=line1.split(",");
        for(i=0;i<arr.length;i++)
        {
                if(arr[5].equals("four"))
                {
                        if(arr[i].equals("?"))
                        {
                                arr[i]=String.valueOf(mean1);
                        }}
                if(arr[5].equals("two"))
                {
                        if(arr[i].equals("?"))
                        {
                                arr[i]=String.valueOf(mean2);
                        }}
                if(arr[5].equals("?"))
                {
                        if(arr[i].equals("?"))
                        {
                                arr[i]=String.valueOf(mean1);
                        }}
                fw.write(arr[i]+",");
        }
        fw.write("\n");
        line1=br1.readLine();
}
br.close();
fw.close();
br1.close(); }}
```

Input:

```
Mean for no of doors are 4: 109.65263157894736
Mean for no of doors are 2: 138.86764705882354
Mean for no of doors are ?: 148.0
```

Output:

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
16	0	109.6526	bmw	gas	std	four	sedan	rwd	front	103.5	189	66.9	55.7	3230	ohc	six
17	0	138.8676	bmw	gas	std	two	sedan	rwd	front	103.5	193.8	67.9	53.7	3380	ohc	six
18	0	109.6526	bmw	gas	std	four	sedan	rwd	front	110	197	70.9	56.3	3505	ohc	six
19	2	121	chevrolet	gas	std	two	hatchback	fwd	front	88.4	141.1	60.3	53.2	1488	l	three
20	1	98	chevrolet	gas	std	two	hatchback	fwd	front	94.5	155.9	63.6	52	1874	ohc	four
21	0	81	chevrolet	gas	std	four	sedan	fwd	front	94.5	158.8	63.6	52	1909	ohc	four
22	1	118	dodge	gas	std	two	hatchback	fwd	front	93.7	157.3	63.8	50.8	1876	ohc	four
23	1	118	dodge	gas	std	two	hatchback	fwd	front	93.7	157.3	63.8	50.8	1876	ohc	four
24	1	118	dodge	gas	turbo	two	hatchback	fwd	front	93.7	157.3	63.8	50.8	2128	ohc	four
25	1	148	dodge	gas	std	four	hatchback	fwd	front	93.7	157.3	63.8	50.6	1967	ohc	four
26	1	148	dodge	gas	std	four	sedan	fwd	front	93.7	157.3	63.8	50.6	1989	ohc	four
27	1	148	dodge	gas	std	four	sedan	fwd	front	93.7	157.3	63.8	50.6	1989	ohc	four
28	1	148	dodge	gas	turbo	109.6526	sedan	fwd	front	93.7	157.3	63.8	50.6	2191	ohc	four

PRACTICAL-4

- **Preprocessing of Noisy Data. Consider the following values for age attribute of total 21 records : 13, 52, 15, 16, 45, 19, 20, 21, 22, 25, 30, 33, 35, 36, 40, 46, 70, 16, 25, 22, 33. Implement smoothing by bin means to smooth these data, using a suitable bin depth.**

Program :-

```
package a;
import java.io.*;
import java.lang.reflect.Array;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collection;
import java.util.Collections;

public class DM_Practical_4 {
	FileInputStream fin;
	FileOutputStream fout;
	BufferedReader buffIn;
	BufferedOutputStream buffOut;
	String data;

	ArrayList<String> dataArray;
	int binSize, nobin;

	public DM_Practical_4() {
		try {
			fin = new FileInputStream(new File("assets/input.data"));
			buffIn = new BufferedReader(new InputStreamReader(fin));
			String[] cols;
			dataArray = new ArrayList<>();
			while ((data = buffIn.readLine()) != null) {

				cols = data.split(", ");
				dataArray.addAll(Arrays.asList(cols));
				// System.out.println(data);
			}
			/*
			 * for (String d : dataArray) { System.out.println(d); }
			 */
			System.out.println("With Sorted");
			{
				ArrayList<Integer> dta = new ArrayList<>();
				for (String d : dataArray)
					dta.add(Integer.valueOf(d));
				Collections.sort(dta);
				dataArray.clear();
				for (Integer i : dta) {
					dataArray.add(String.valueOf(i));
				}
			}
			// Collections.sort(dataArray);
			for (String d : dataArray) {
				System.out.println(d);
			}
```

```
                buffIn.close();
                fin.close();
            } catch (Exception e) {
                e.getMessage();
            }

            calculateBinSize();
            nobin = dataArray.size() / binSize;
            System.out.println("Total Size :" + dataArray.size() + "Bin Size :" + binSize);
            normlizeBins(0);
        }

        private void normlizeBins(int choice) {
            // ArrayList<Integer> meanArray = new ArrayList<>();
            int sum = 0;
            if (choice == 0) {
                for (int i = 0; i < nobin; i++) {
                    for (int j = 0; j < binSize; j++) {
                        // meanArray.add(Integer.valueOf(dataArray.get(i + j)));
                        sum += Integer.valueOf(dataArray.get(i * binSize + j));
                    }
                    for (int k = 0; k < binSize; k++) {
                        dataArray.set(i * binSize + k, String.valueOf((double) sum / (double) binSize));
                    }
                    sum = 0;
                }
            }
            for (String d : dataArray) {
                System.out.println(d);
            }
            // TODO Auto-generated method stub

        }

        private void calculateBinSize() {
            int min = dataArray.size();
            for (int i = 1; i < dataArray.size() / 2; i++) {
                if (dataArray.size() % i == 0) {
                    if (min > Math.abs(dataArray.size() / i - i)) {
                        min = Math.abs(dataArray.size() / i) - i;
                        System.out.println("bin : " + binSize + "min : " + (Math.abs(dataArray.size() / i) - i));
                        binSize = i;
                    }
                }
            }
        }

        public static void main(String[] args) {
            new DM_Practical_4();
        }
}
```

Output :-

```
Total Size :21Bin Size :3
14.666666666666666
14.666666666666666
14.666666666666666
18.333333333333332
18.333333333333332
18.333333333333332
21.666666666666668
21.666666666666668
21.666666666666668
26.666666666666668
26.666666666666668
26.666666666666668
33.666666666666664
33.666666666666664
33.666666666666664
40.333333333333336
40.333333333333336
40.333333333333336
56.0
56.0
56.0
```

PRACTICAL-5

- **Preprocessing of Noisy Data. Consider the following values for age attribute of total 21 records : 13, 52, 15, 16, 45, 19, 20, 21, 22, 25, 30, 33, 35, 36, 40, 46, 70, 16, 25, 22, 33. Implement smoothing by bin medians to smooth these data, using a suitable bin depth.**

Program :-

```
package a;
import java.io.*;
import java.lang.reflect.Array;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collection;
import java.util.Collections;

public class DM_Practical_5 {
	FileInputStream fin;
	FileOutputStream fout;
	BufferedReader buffIn;
	BufferedOutputStream buffOut;
	String data;
	ArrayList<String> dataArray;
	int binSize, nobin;

	public DM_Practical_5() {
		try {
			fin = new FileInputStream(new File("assets/input.data"));
			buffIn = new BufferedReader(new InputStreamReader(fin));
			String[] cols;
			dataArray = new ArrayList<>();
			while ((data = buffIn.readLine()) != null) {

				cols = data.split(", ");
				dataArray.addAll(Arrays.asList(cols));                    }
			System.out.println("With Sorted");
			{
				ArrayList<Integer> dta = new ArrayList<>();
				for (String d : dataArray)
					dta.add(Integer.valueOf(d));
				Collections.sort(dta);
				dataArray.clear();
				for (Integer i : dta) {
					dataArray.add(String.valueOf(i));
				}
			}
			for (String d : dataArray) {
				System.out.println(d);
			}
			buffIn.close();
			fin.close();
		} catch (Exception e) {
			e.getMessage();
		}
```

```
                calculateBinSize();
                nobin = dataArray.size() / binSize;
                System.out.println("Total Size :" + dataArray.size() + "Bin Size :" + binSize);
                normlizeBins(1);
        }

        private void normlizeBins(int choice) {
                // ArrayList<Integer> meanArray = new ArrayList<>();
                int sum = 0;
                if (choice == 1) {
                        for (int i = 0; i < nobin; i++) {

                                        if (binSize % 2 == 0) {

        sum=Integer.valueOf(dataArray.get(i*binSize+binSize/2))+Integer.valueOf(dataArray.get(i*binSize+((binSize/2)-1)));
                                                sum=sum/2;
                                        } else {
                                                sum=Integer.valueOf(dataArray.get(i*binSize+binSize/2));
                                        }
                                for (int k = 0; k < binSize; k++) {
                                        dataArray.set(i * binSize + k, String.valueOf(sum));
//                                      System.out.print(dataArray.get(i*binSize+k)+" ");
                                }
                                sum = 0;
                        }
                }
                for (String d : dataArray) {
                        System.out.println(d);
                }
        }
        private void calculateBinSize() {
                int min = dataArray.size();
                for (int i = 1; i < dataArray.size() / 2; i++) {
                        if (dataArray.size() % i == 0) {
                                if (min > Math.abs(dataArray.size() / i - i)) {
                                        min = Math.abs(dataArray.size() / i) - i;
                                        System.out.println("bin : " + binSize + "min : " + (Math.abs(dataArray.size() / i) - i));
                                        binSize = i;
                                }       }
                }
        }

        public static void main(String[] args) {
                new DM_Practical_5();
        }
}
```

Output :-

```
Total Size :21Bin Size :3
15 15 15
19 19 19
22 22 22
25 25 25
33 33 33
40 40 40
52 52 52
```

PRACTICAL-6

- **Preprocessing of Noisy Data. Consider the following values for age attribute of total 21 records : 13, 52, 15, 16, 45, 19, 20, 21, 22, 25, 30, 33, 35, 36, 40, 46, 70, 16, 25, 22, 33. Implement smoothing by bin boundaries to smooth these data, using a suitable bin depth.**

Program :-

```
package a;
import java.io.*;
import java.lang.reflect.Array;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collection;
import java.util.Collections;

public class DM_Practical_6 {
	FileInputStream fin;
	FileOutputStream fout;
	BufferedReader buffIn;
	BufferedOutputStream buffOut;
	String data;

	ArrayList<String> dataArray;
	int binSize, nobin;

	public DM_Practical_6() {
		try {
			fin = new FileInputStream(new File("assets/input.data"));
			buffIn = new BufferedReader(new InputStreamReader(fin));
			String[] cols;
			dataArray = new ArrayList<>();
			while ((data = buffIn.readLine()) != null) {

				cols = data.split(", ");
				dataArray.addAll(Arrays.asList(cols));
				// System.out.println(data);
			}
			System.out.println("With Sorted");
			{
				ArrayList<Integer> dta = new ArrayList<>();
				for (String d : dataArray)
					dta.add(Integer.valueOf(d));
				Collections.sort(dta);
				dataArray.clear();
				for (Integer i : dta) {
					dataArray.add(String.valueOf(i));
				}
			}
			for (String d : dataArray) {
				System.out.println(d);
			}

			buffIn.close();
			fin.close();
		} catch (Exception e) {
			e.getMessage();
```

```
		}

		calculateBinSize();
		nobin = dataArray.size() / binSize;
		System.out.println("Total Size :" + dataArray.size() + "Bin Size :" + binSize);
		normlizeBins(2);
	}

	private void normlizeBins(int choice) {
		// ArrayList<Integer> meanArray = new ArrayList<>();
		int sum = 0;
		if (choice == 2) {
			int a = 0, min = 0, max = 0;
			for (int i = 0; i < nobin; i++) {
				min = Integer.valueOf(dataArray.get(i * binSize + 0));
				max = Integer.valueOf(dataArray.get(i * binSize + (binSize-1)));
				System.out.print(dataArray.get(i*binSize)+" ");
				for (int j = 1; j < binSize-1; j++) {
					a = Integer.valueOf(dataArray.get(i * binSize + j));
					if (Math.abs(a - min) < Math.abs(a - max))
						dataArray.set(i * binSize + j, dataArray.get(i * binSize + 0));
					else
						dataArray.set(i * binSize + j, dataArray.get(i * binSize + (binSize-
1)));

					System.out.print(dataArray.get(i*binSize+j)+" ");
				}
				System.out.println(dataArray.get(i*binSize+(binSize-1))+" ");

			}
		}
	}
	private void calculateBinSize() {
		int min = dataArray.size();
		for (int i = 1; i < dataArray.size() / 2; i++) {
			if (dataArray.size() % i == 0) {
				if (min > Math.abs(dataArray.size() / i - i)) {
					min = Math.abs(dataArray.size() / i) - i;
					System.out.println("bin : " + binSize + "min : " + (Math.abs(dataArray.size() / i) - i));
					binSize = i;
				}
			}
		}
}
	public static void main(String[] args) {
		new DM_Practical_6();
	}
}
```

Output :-

```
Total Size :21Bin Size :3
13 16 16
16 20 20
21 22 22
25 25 30
33 33 35
36 36 45
46 46 70
```