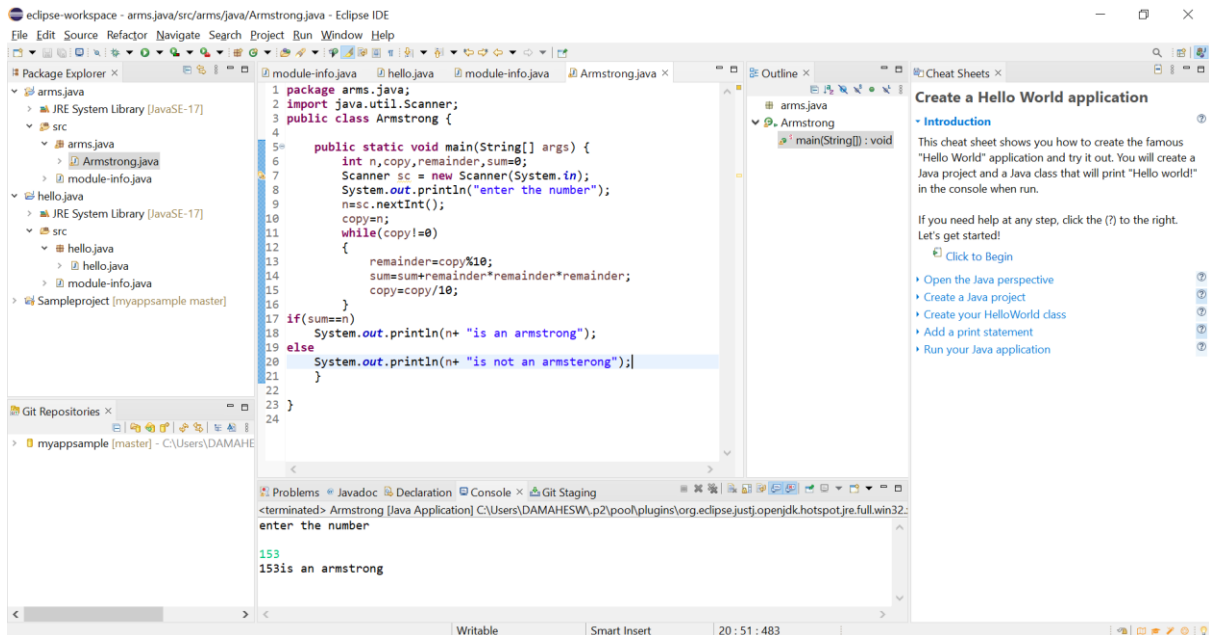


CORE JAVA

ASSIGNMENT – 1

1) Find out if the given number is an Armstrong number.

Logic: -if 153 is the Supplied value, then $1^3 + 5^3 + 3^3 = 1 + 125 + 27 = 153$ This is the same as supplied value hence it is an Armstrong number.

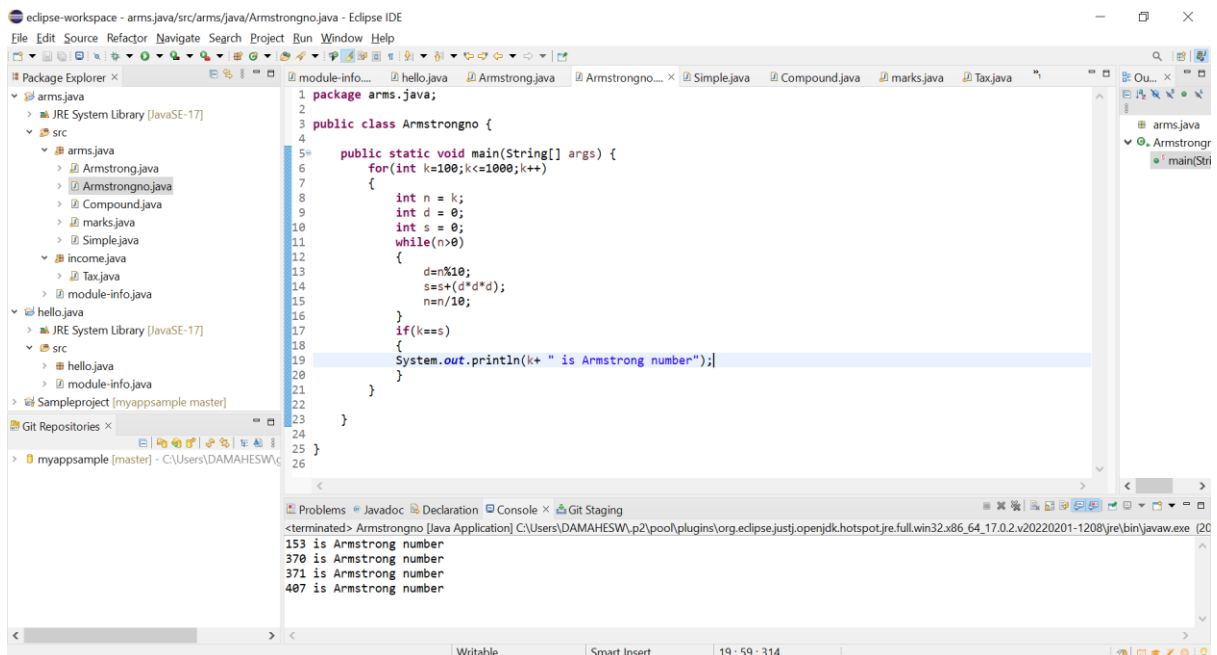


```
1 package arms.java;
2 import java.util.Scanner;
3 public class Armstrong {
4
5     public static void main(String[] args) {
6         int n,copy,remainder,sum=0;
7         Scanner sc = new Scanner(System.in);
8         System.out.println("enter the number");
9         n=sc.nextInt();
10        copy=n;
11        while(copy!=0)
12        {
13            remainder=copy%10;
14            sum=sum+remainder*remainder*remainder;
15            copy=copy/10;
16        }
17        if(sum==n)
18            System.out.println(n+ " is an armstrong");
19        else
20            System.out.println(n+ " is not an armstrong");
21    }
22 }
23
24
```

Console Output:

```
<terminated> Armstrong [Java Application] C:\Users\DAMAHESW\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32-
enter the number
153
153 is an armstrong
```

2. Find out all the Armstrong numbers falling in the range of 100-999.



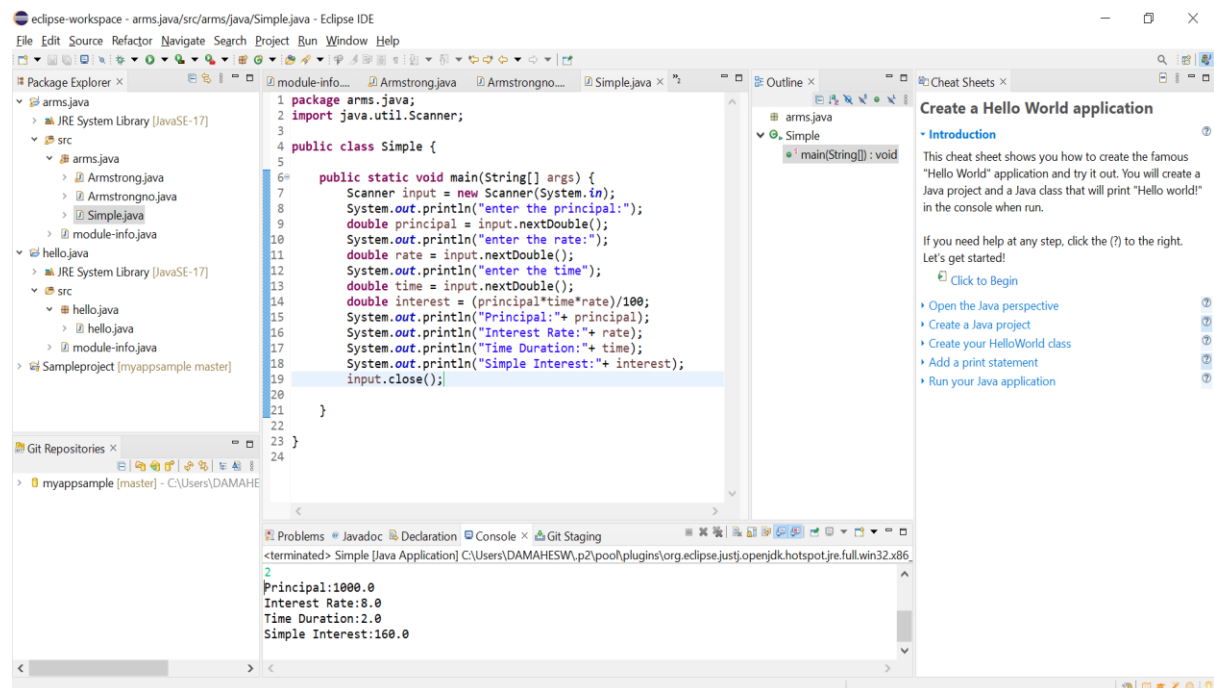
```
1 package arms.java;
2
3 public class Armstrongno {
4
5     public static void main(String[] args) {
6         for(int k=100;k<=1000;k++)
7         {
8             int n = k;
9             int d = 0;
10            int s = 0;
11            while(n>0)
12            {
13                d=n%10;
14                s=s+(d*d*d);
15                n=n/10;
16            }
17            if(k==s)
18            {
19                System.out.println(k+ " is Armstrong number");
20            }
21        }
22    }
23 }
24
25
26
```

Console Output:

```
<terminated> Armstrongno [Java Application] C:\Users\DAMAHESW\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32-x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe (20
153 is Armstrong number
370 is Armstrong number
371 is Armstrong number
407 is Armstrong number
```

3. Find out the simple as well as the compound interest of supplied value.

Simple Interest:



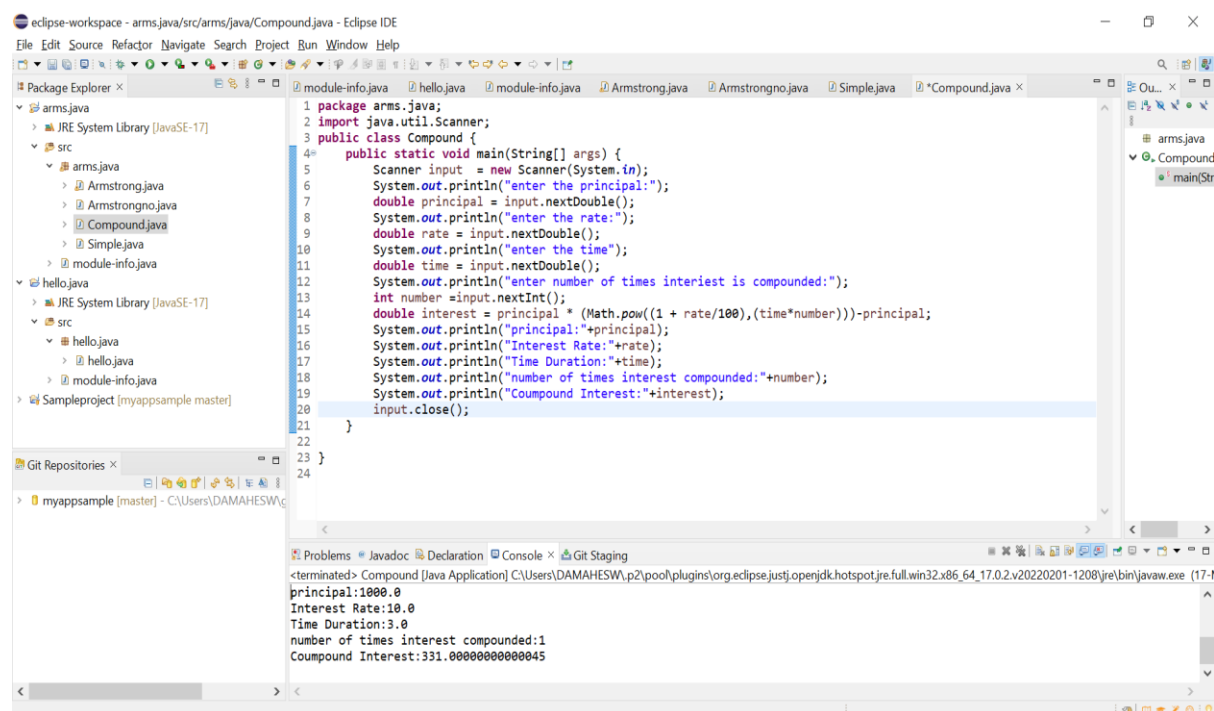
The screenshot shows the Eclipse IDE with the Simple.java file open. The code calculates simple interest based on user input for principal, rate, and time. The console output shows the results of the program execution.

```
1 package arms.java;
2 import java.util.Scanner;
3
4 public class Simple {
5
6     public static void main(String[] args) {
7         Scanner input = new Scanner(System.in);
8         System.out.println("enter the principal:");
9         double principal = input.nextDouble();
10        System.out.println("enter the rate:");
11        double rate = input.nextDouble();
12        System.out.println("enter the time");
13        double time = input.nextDouble();
14        double interest = (principal*time*rate)/100;
15        System.out.println("Principal:"+ principal);
16        System.out.println("Interest Rate:"+ rate);
17        System.out.println("Time Duration:"+ time);
18        System.out.println("Simple Interest:"+ interest);
19        input.close();
20    }
21 }
22
23 }
24 }
```

Console Output:

```
<terminated> Simple [Java Application] C:\Users\DAMAHESW\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe (17-0)
Principal:1000.0
Interest Rate:8.0
Time Duration:2.0
Simple Interest:160.0
```

Compound Interest:



The screenshot shows the Eclipse IDE with the Compound.java file open. The code calculates compound interest based on user input for principal, rate, time, and the number of times interest is compounded. The console output shows the results of the program execution.

```
1 package arms.java;
2 import java.util.Scanner;
3 public class Compound {
4     public static void main(String[] args) {
5         Scanner input = new Scanner(System.in);
6         System.out.println("enter the principal:");
7         double principal = input.nextDouble();
8         System.out.println("enter the rate:");
9         double rate = input.nextDouble();
10        System.out.println("enter the time");
11        double time = input.nextDouble();
12        System.out.println("enter number of times interest is compounded:");
13        int number = input.nextInt();
14        double interest = principal * (Math.pow((1 + rate/100),(time*number)))-principal;
15        System.out.println("Principal:"+principal);
16        System.out.println("Interest Rate:"+rate);
17        System.out.println("Time Duration:"+time);
18        System.out.println("number of times interest compounded:"+number);
19        System.out.println("Compound Interest:"+interest);
20        input.close();
21    }
22 }
23 }
24 }
```

Console Output:

```
<terminated> Compound [Java Application] C:\Users\DAMAHESW\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe (17-0)
Principal:1000.0
Interest Rate:10.0
Time Duration:3.0
number of times interest compounded:1
Compound Interest:331.000000000000045
```

4. Supply marks of three subject and declare the result, result declaration is based on below conditions:

Condition 1: All subjects marks is greater than 60 is Passed

Condition 2: Any two subjects marks are greater than 60 is Promoted

Condition 3: Any one subject mark is greater than 60 or all subjects' marks less than 60 is failed.

The screenshot shows the Eclipse IDE with a project named 'arms.java'. The main editor displays the following Java code:

```
5 public class marks {
6
7     public static void main(String[] args) {
8         int m1,m2,m3;
9         Scanner sc = new Scanner(System.in);
10
11         System.out.println("enter the first subjects marks");
12         m1=sc.nextInt();
13
14         System.out.println("enter the second subject marks");
15         m2=sc.nextInt();
16
17         System.out.println("enter the third subject marks");
18         m3=sc.nextInt();
19
20         if((m1>60)&&(m2>60)&&(m3>60))
21         {
22             System.out.println("result is passed");
23         }
24         else if ((m1>60&& m2>60)|| (m2>60&&m3>60)|| (m1>60&&m3>60)) {
25             System.out.println(" result is:promoted");
26         }
27         else {
28             System.out.println("result is:failed");
29         }
30     }
31 }
```

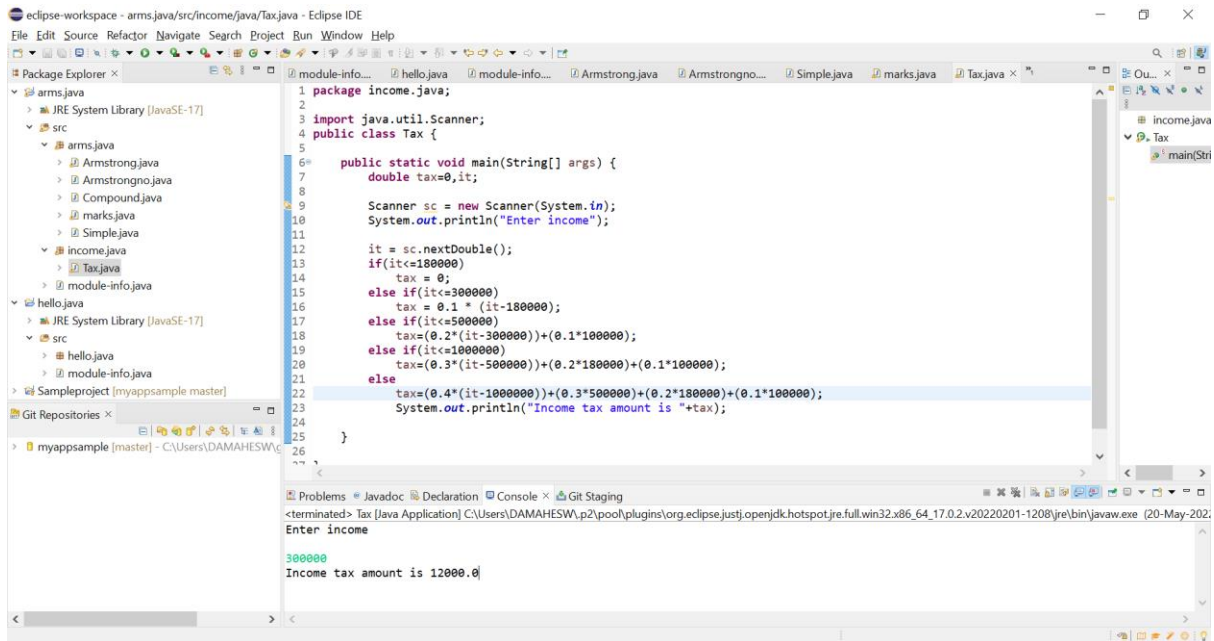
The Package Explorer on the left shows the project structure with 'arms.java' and its sub-packages. The Console at the bottom shows the output of the program:

```
<terminated> marks [Java Application] C:\Users\DAMAHE...
60
enter the third subject marks
58
result is:failed
```

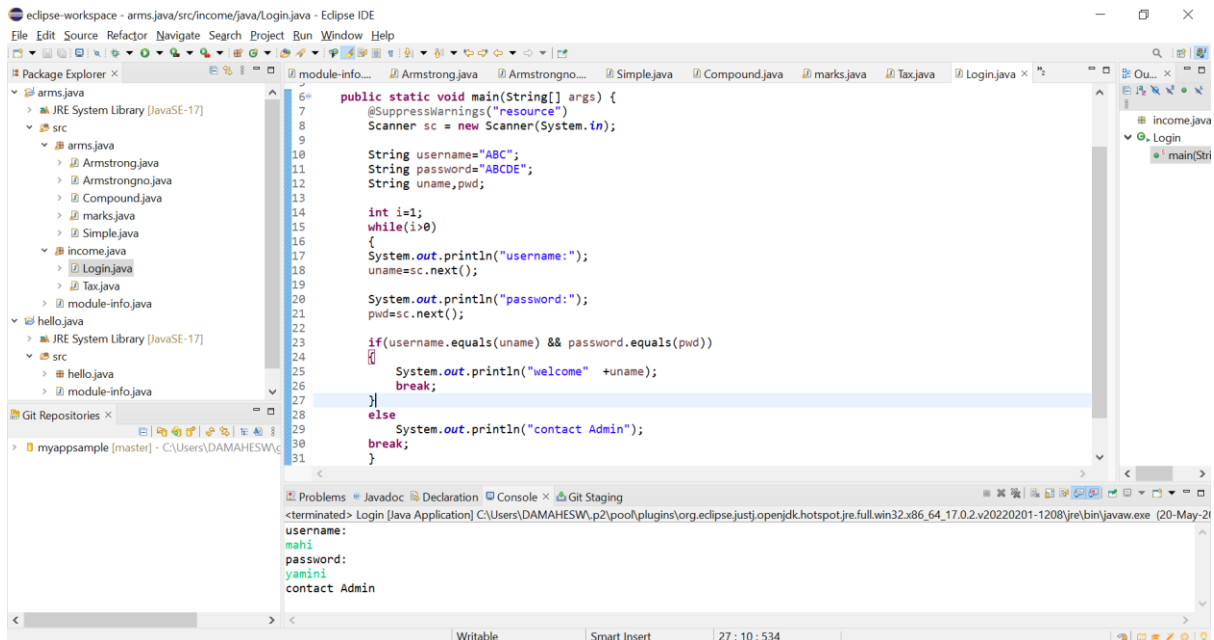
5. Calculate the income tax on the basis of following table.

Note:-Assume slab is consider for Male, Female as well as Senior citizen

Slab	Income Range	Tax payable in Percentage
Slab A	0-1,80,000	Nil
Slab B	1,81,001-3,00,000	10%
Slab C	3,00,001-5,00,000	20%
Slab D	5,00,001-10,00,000	30%



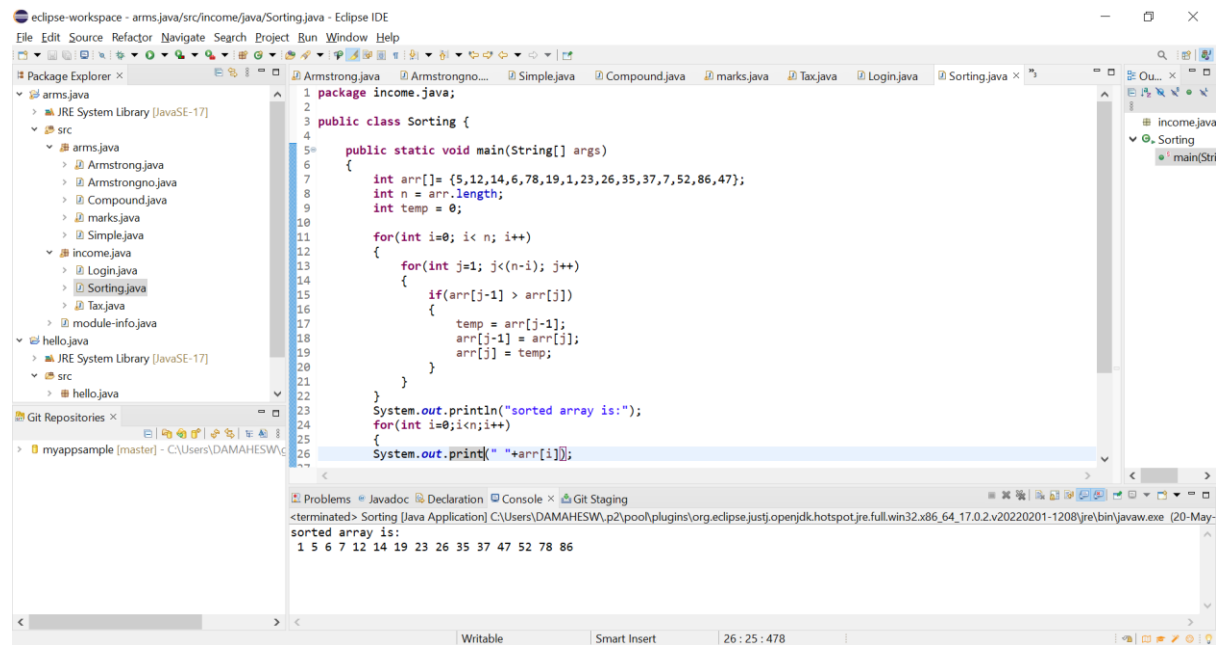
6. Consider a CUI based application, where you are asking a user to enter his Login name and password, after entering the valid user-id and password it will print the message “Welcome” along with user name. As per the validation is concerned, the program should keep a track of login attempts. After three attempts a message should be flashed saying “Contact Admin” and the program should terminate.



7. There is an Array which is of the size 15, which may or may not be sorted. You should write a program to accept a number and search if it is contained in the array.

Example: 51214678191232635377528647

Value to be search is 19.



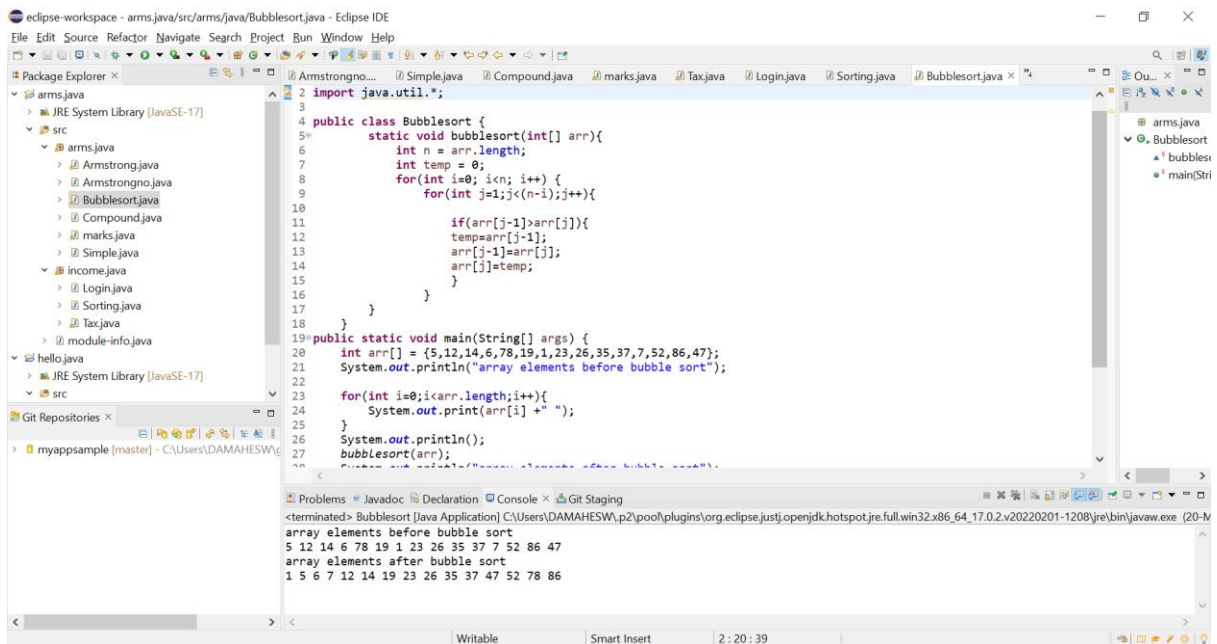
The screenshot shows the Eclipse IDE with a Java project named 'arms.java'. The 'Package Explorer' on the left shows the project structure, including a 'src' folder with files like 'Armstrong.java', 'Simple.java', 'Compound.java', 'marks.java', 'Tax.java', 'Login.java', 'Sorting.java', and 'module-info.java'. The 'Sorting.java' file is open in the editor, showing the following code:

```
1 package income.java;
2
3 public class Sorting {
4
5     public static void main(String[] args)
6     {
7         int arr[] = {5,12,14,6,78,19,1,23,26,35,37,7,52,86,47};
8         int n = arr.length;
9         int temp = 0;
10
11         for(int i=0; i<n; i++)
12         {
13             for(int j=1; j<(n-i); j++)
14             {
15                 if(arr[j-1] > arr[j])
16                 {
17                     temp = arr[j-1];
18                     arr[j-1] = arr[j];
19                     arr[j] = temp;
20                 }
21             }
22         }
23         System.out.println("sorted array is:");
24         for(int i=0; i<n; i++)
25         {
26             System.out.print(" "+arr[i]);
27         }
28     }
29 }
```

The 'Console' at the bottom shows the output of the program:

```
<terminated> Sorting [Java Application] C:\Users\DAMAHESW\p2\poo\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe (20-May-2022 10:10:10 AM)
sorted array is:
1 5 6 7 12 14 19 23 26 35 37 47 52 78 86
```

8. Using the above table write method apply sorting using Bubble Sort.

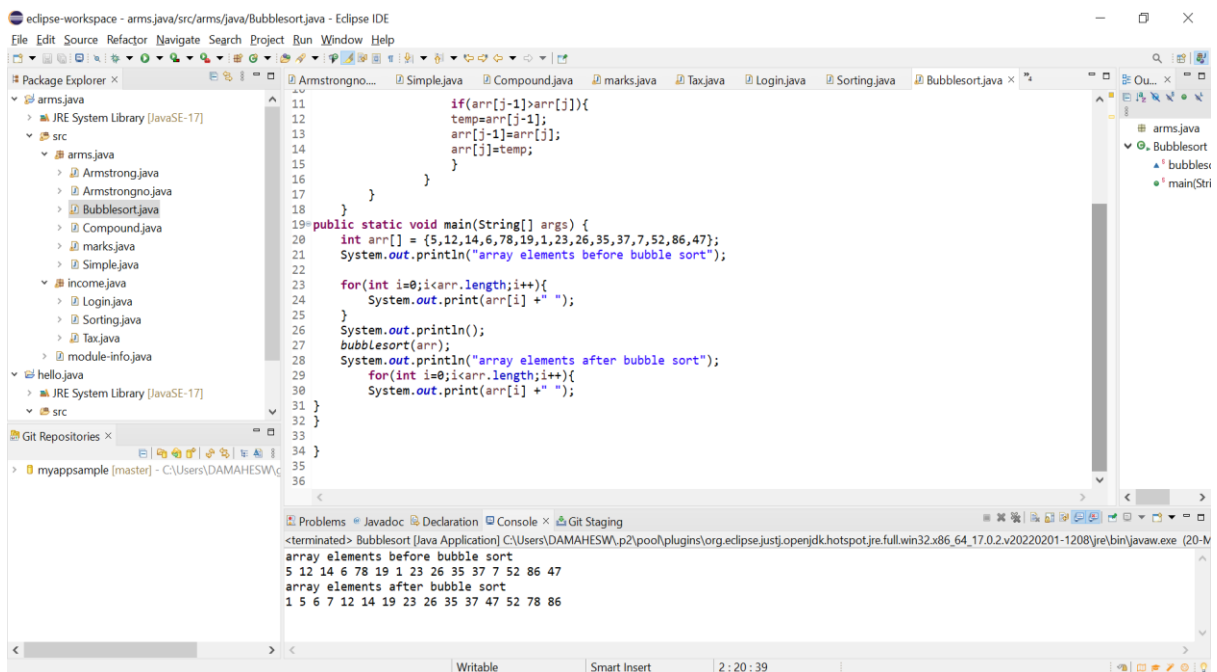


```
import java.util.*;

public class Bubblesort {
    static void bubblesort(int[] arr){
        int n = arr.length;
        int temp = 0;
        for(int i=0; i<n; i++){
            for(int j=1; j<(n-i); j++){
                if(arr[j-1]>arr[j]){
                    temp=arr[j-1];
                    arr[j-1]=arr[j];
                    arr[j]=temp;
                }
            }
        }
    }

    public static void main(String[] args) {
        int arr[] = {5,12,14,6,78,19,1,23,26,35,37,7,52,86,47};
        System.out.println("array elements before bubble sort");
        for(int i=0; i<arr.length; i++){
            System.out.print(arr[i] + " ");
        }
        System.out.println();
        bubblesort(arr);
        System.out.println("array elements after bubble sort");
    }
}
```

array elements before bubble sort
5 12 14 6 78 19 1 23 26 35 37 7 52 86 47
array elements after bubble sort
1 5 6 7 12 14 19 23 26 35 37 47 52 78 86

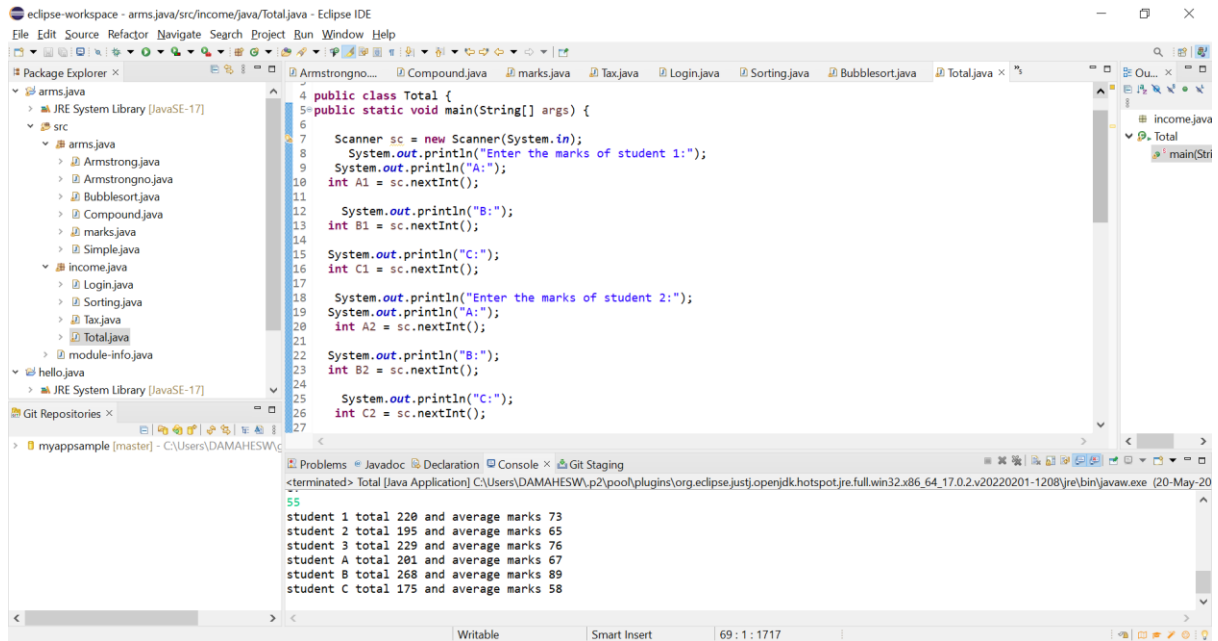


```
if(arr[j-1]>arr[j]){
    temp=arr[j-1];
    arr[j-1]=arr[j];
    arr[j]=temp;
}
}

public static void main(String[] args) {
    int arr[] = {5,12,14,6,78,19,1,23,26,35,37,7,52,86,47};
    System.out.println("array elements before bubble sort");
    for(int i=0; i<arr.length; i++){
        System.out.print(arr[i] + " ");
    }
    System.out.println();
    bubblesort(arr);
    System.out.println("array elements after bubble sort");
    for(int i=0; i<arr.length; i++){
        System.out.print(arr[i] + " ");
    }
}
```

array elements before bubble sort
5 12 14 6 78 19 1 23 26 35 37 7 52 86 47
array elements after bubble sort
1 5 6 7 12 14 19 23 26 35 37 47 52 78 86

9) Accept the marks of three students for the subject say A, B, C. Find the total scored and the average in all the subject. Also find the total and average scored by students in each respective subject.



The screenshot shows the Eclipse IDE interface. The Package Explorer on the left displays a project structure with folders 'arms.java', 'income.java', and 'hello.java'. The 'arms.java' folder contains subfolders 'src' and 'JRE System Library [JavaSE-17]'. The 'src' folder contains files 'Armstrong.java', 'Armstrongno.java', 'Bubblesort.java', 'Compound.java', 'marks.java', 'Simple.java', 'Login.java', 'Sorting.java', 'Tax.java', and 'Total.java'. The 'income.java' folder contains 'module-info.java'. The 'hello.java' folder is empty. The main editor window shows the code for 'Total.java'. The code is as follows:

```
4 public class Total {
5     public static void main(String[] args) {
6
7         Scanner sc = new Scanner(System.in);
8         System.out.println("Enter the marks of student 1:");
9         System.out.println("A:");
10        int A1 = sc.nextInt();
11
12        System.out.println("B:");
13        int B1 = sc.nextInt();
14
15        System.out.println("C:");
16        int C1 = sc.nextInt();
17
18        System.out.println("Enter the marks of student 2:");
19        System.out.println("A:");
20        int A2 = sc.nextInt();
21
22        System.out.println("B:");
23        int B2 = sc.nextInt();
24
25        System.out.println("C:");
26        int C2 = sc.nextInt();
27    }
28 }
```

The Console window at the bottom shows the output of the program:

```
<terminated> Total [Java Application] C:\Users\DAMAHESW\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe (20-May-20
55
student 1 total 220 and average marks 73
student 2 total 195 and average marks 65
student 3 total 229 and average marks 76
student A total 201 and average marks 67
student B total 268 and average marks 89
student C total 175 and average marks 58
```


eclipse-workspace - arms.java/src/income/java/Total.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer

- arms.java
 - JRE System Library [JavaSE-17]
 - src
 - arms.java
 - Armstrong.java
 - Armstrongno.java
 - Bubblesort.java
 - Compound.java
 - marks.java
 - Simple.java
 - income.java
 - Login.java
 - Sorting.java
 - Tax.java
 - Total.java
 - module-info.java
- hello.java
 - JRE System Library [JavaSE-17]
- Git Repositories
 - myappsample [master] - C:\Users\DAMAHESW\...

Armstrongno... Compound.java marks.java Tax.java Login.java Sorting.java Bubblesort.java Total.java

```
27
28 System.out.println("Enter the marks of student 3:");
29 System.out.println("A:");
30 int A3 = sc.nextInt();
31
32 System.out.println("B:");
33 int B3 = sc.nextInt();
34
35 System.out.println("C:");
36 int C3 = sc.nextInt();
37
38 int sum1 = A1+B1+C1;
39 int sum2 = A2+B2+C2;
40 int sum3 = A3+B3+C3;
41
42 int Avg1 = sum1/3;
43 int Avg2 = sum2/3;
44 int Avg3 = sum3/3;
45
46 int A_subTotal = A1+A2+A3;
47 int B_subTotal = B1+B2+B3;
48 int C_subTotal = C1+C2+C3;
49
50 int A Avg = A_subTotal/3;
```

Problems Javadoc Declaration Console Git Staging

<terminated> Total [Java Application] C:\Users\DAMAHESW\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe (20-May-2022)

```
55
56 student 1 total 220 and average marks 73
57 student 2 total 195 and average marks 65
58 student 3 total 229 and average marks 76
59 student A total 201 and average marks 67
60 student B total 268 and average marks 89
61 student C total 175 and average marks 58
```

Writable Smart Insert 69 : 1 : 1717

eclipse-workspace - arms.java/src/income/java/Total.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Package Explorer

- arms.java
 - JRE System Library [JavaSE-17]
 - src
 - arms.java
 - Armstrong.java
 - Armstrongno.java
 - Bubblesort.java
 - Compound.java
 - marks.java
 - Simple.java
 - income.java
 - Login.java
 - Sorting.java
 - Tax.java
 - Total.java
 - module-info.java
- hello.java
 - JRE System Library [JavaSE-17]
- Git Repositories
 - myappsample [master] - C:\Users\DAMAHESW\...

Armstrongno... Compound.java marks.java Tax.java Login.java Sorting.java Bubblesort.java Total.java

```
46 int A_subTotal = A1+A2+A3;
47 int B_subTotal = B1+B2+B3;
48 int C_subTotal = C1+C2+C3;
49
50 int A_Avg = A_subTotal/3;
51 int B_Avg = B_subTotal/3;
52 int C_Avg = C_subTotal/3;
53
54 System.out.println("student 1 total " +sum1+ " and average marks "+Avg1);
55 System.out.println("student 2 total " +sum2+ " and average marks "+Avg2);
56 System.out.println("student 3 total " +sum3+ " and average marks "+Avg3);
57
58 System.out.println("student A total " +A_subTotal+ " and average marks "+A_Avg);
59 System.out.println("student B total " +B_subTotal+ " and average marks "+B_Avg);
60 System.out.println("student C total " +C_subTotal+ " and average marks "+C_Avg);
61
62 }
63
64
65 }
66
67
68
69
```

Problems Javadoc Declaration Console Git Staging

<terminated> Total [Java Application] C:\Users\DAMAHESW\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bin\javaw.exe (20-May-2022)

```
55
56 student 1 total 220 and average marks 73
57 student 2 total 195 and average marks 65
58 student 3 total 229 and average marks 76
59 student A total 201 and average marks 67
60 student B total 268 and average marks 89
61 student C total 175 and average marks 58
```

Writable Smart Insert 69 : 1 : 1717