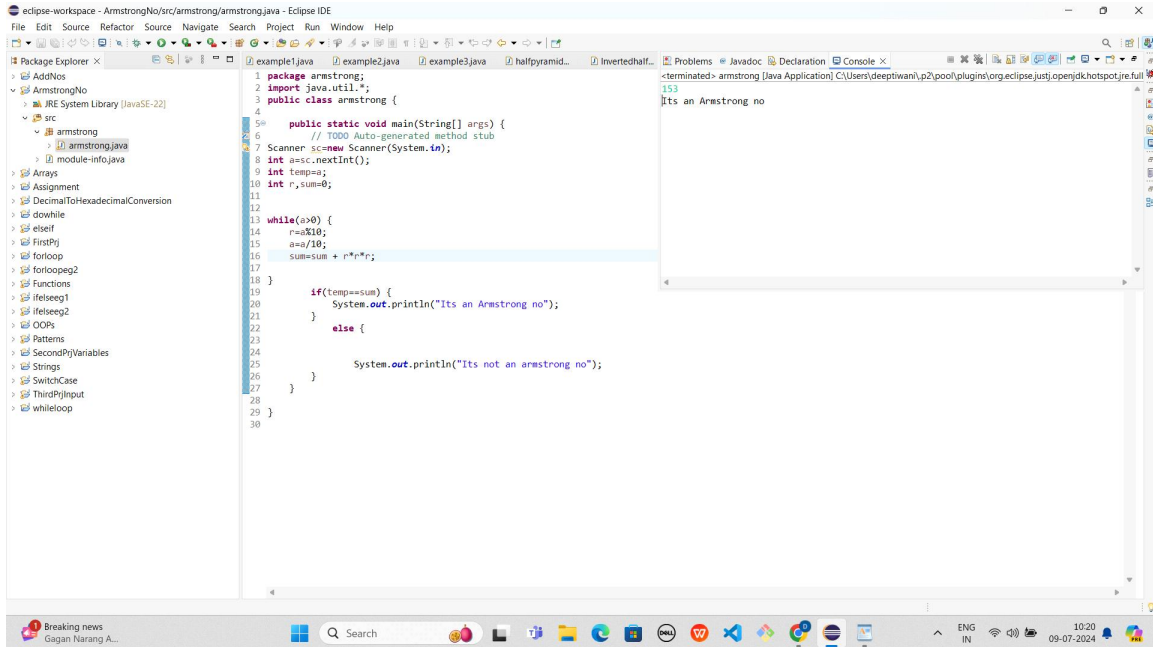


Verinite Technologies Pvt Ltd

Core Java Assignments

Program to Check Armstrong Number



The screenshot shows the Eclipse IDE interface. The Package Explorer on the left lists a project named 'ArmstrongNo' with a source folder 'src' containing 'armstrong.java'. The main editor displays the code for 'armstrong.java'. The code imports 'java.util.*' and uses a 'Scanner' to take input from the user. It then checks if the input number is an Armstrong number by calculating the sum of the cubes of its digits. The console on the right shows the output: '153' followed by 'Its an Armstrong no'.

```
1 package armstrong;
2 import java.util.*;
3 public class armstrong {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         Scanner sc=new Scanner(System.in);
8         int a=sc.nextInt();
9         int temp=a;
10        int r,sum=0;
11
12        while(a>0) {
13            r=a%10;
14            a=a/10;
15            sum=sum + r*r*r;
16        }
17
18        if(temp==sum) {
19            System.out.println("Its an Armstrong no");
20        }
21        else {
22
23            System.out.println("Its not an armstrong no");
24        }
25    }
26 }
27
28
29 }
30 }
```

Console Output:

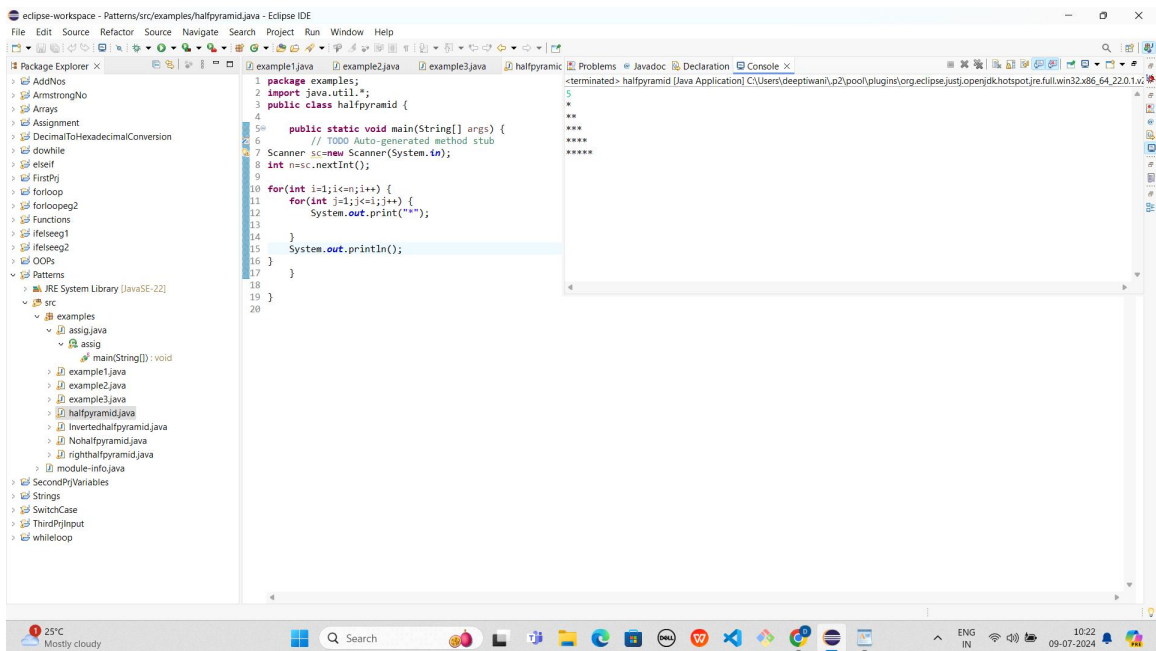
```
<terminated> armstrong [Java Application] C:\Users\deepthi\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full
153
Its an Armstrong no
```

Program to Print Right Triangle Star Pattern

Input : n = 5

Output:

```
*
* *
* * *
* * * *
* * * * *
```



The screenshot shows the Eclipse IDE interface. The Package Explorer on the left lists the project structure, including the 'halfpyramid.java' file. The main editor displays the following Java code:

```
1 package examples;
2 import java.util.*;
3 public class halfpyramid {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         Scanner sc=new Scanner(System.in);
8         int n=sc.nextInt();
9
10        for(int i=1;i<=n;i++) {
11            for(int j=1;j<=i;j++) {
12                System.out.print("*");
13            }
14            System.out.println();
15        }
16    }
17 }
18
19
20 }
```

The Console window on the right shows the output of the program, which matches the expected right triangle star pattern for n=5:

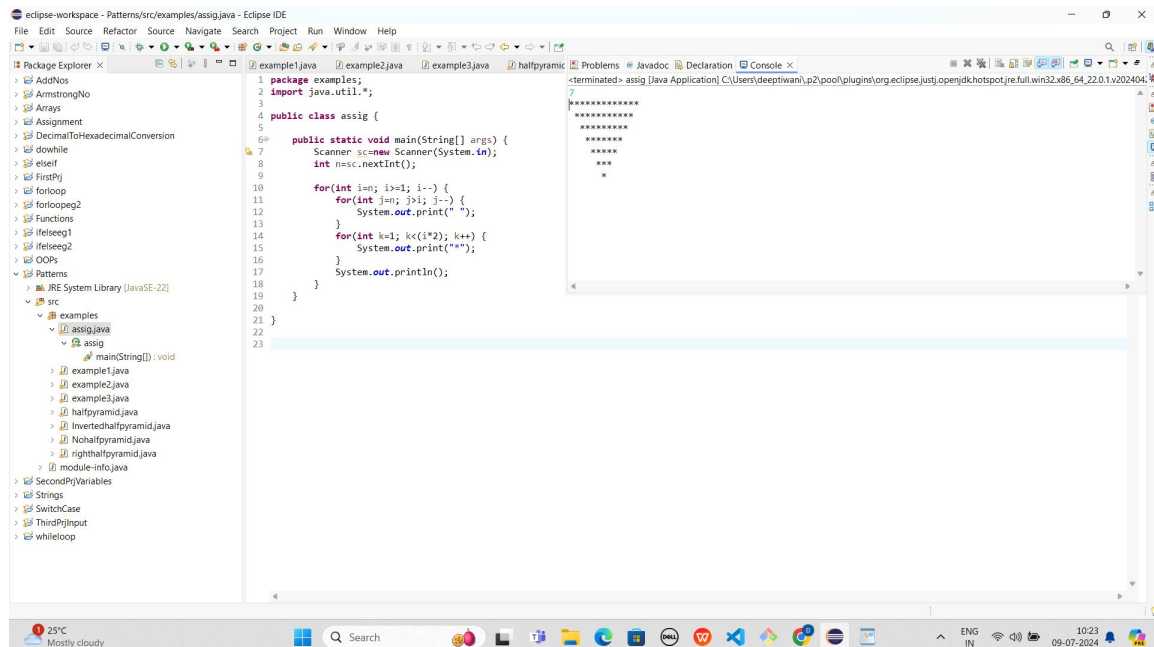
```
<terminated> halfpyramid [Java Application] C:\Users\deeptiwan\p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.22.0.1.v...
```

Program to Print Reverse Pyramid Star Pattern

Input: number = 7

Output:

```
*****
*****
*****
*****
*****
***
*
```



The screenshot shows the Eclipse IDE with a Java project named 'Patterns'. The 'Package Explorer' on the left shows a package 'examples' containing several Java files, including 'assign.java'. The 'Main' window displays the code for 'assign.java', which is a Java program to print a reverse pyramid star pattern. The code uses a Scanner to take input from the user, which is 7. It then uses nested loops to print the pattern. The 'Console' window on the right shows the output of the program, which is a reverse pyramid star pattern with 7 rows.

```
1 package examples;
2 import java.util.*;
3
4 public class assign {
5
6     public static void main(String[] args) {
7         Scanner sc=new Scanner(System.in);
8         int n=sc.nextInt();
9
10        for(int i=n; i>=1; i--) {
11            for(int j=1; j>=i; j++) {
12                System.out.print(" ");
13            }
14            for(int k=1; k<=((i*2)); k++) {
15                System.out.print("*");
16            }
17            System.out.println();
18        }
19    }
20 }
21
22
23
```

Program For Decimal to Hexadecimal Conversion

Input : 10

Output: A

Input : 2545

Output: 9F1

The screenshot shows the Eclipse IDE with a Java project named 'DecimalToHexadecimalConversion'. The package explorer on the left shows the project structure. The main editor displays the source code for 'decimaltohexadecimal.java'. The code is as follows:

```
1 package decimaltohexadecimal;
2 import java.util.*;
3 public class decimaltohexadecimal {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         Scanner sc=new Scanner(System.in);
8         int decimal;
9         int remainder;
10        String hexadecimal="";
11        char hex[]={'0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'};
12        System.out.print("Input a decimal no:");
13        decimal=sc.nextInt();
14        while(decimal>0) {
15            remainder=decimal%16;
16            hexadecimal=hex[remainder]+hexadecimal;
17            decimal=decimal/16;
18        }
19        System.out.print("Hexadecimal value is:"+hexadecimal);
20    }
21 }
22 }
23 }
```

The console on the right shows the output: "Input a decimal no:10" and "Hexadecimal value is:A".

The screenshot shows the Eclipse IDE with the same Java project. The main editor displays the source code for 'decimaltohexadecimal.java'. The code is as follows:

```
1 package decimaltohexadecimal;
2 import java.util.*;
3 public class decimaltohexadecimal {
4
5     public static void main(String[] args) {
6         // TODO Auto-generated method stub
7         Scanner sc=new Scanner(System.in);
8         int decimal;
9         int remainder;
10        String hexadecimal="";
11        char hex[]={'0','1','2','3','4','5','6','7','8','9','A','B','C','D','E','F'};
12        System.out.print("Input a decimal no:");
13        decimal=sc.nextInt();
14        while(decimal>0) {
15            remainder=decimal%16;
16            hexadecimal=hex[remainder]+hexadecimal;
17            decimal=decimal/16;
18        }
19        System.out.print("Hexadecimal value is:"+hexadecimal);
20    }
21 }
22 }
23 }
```

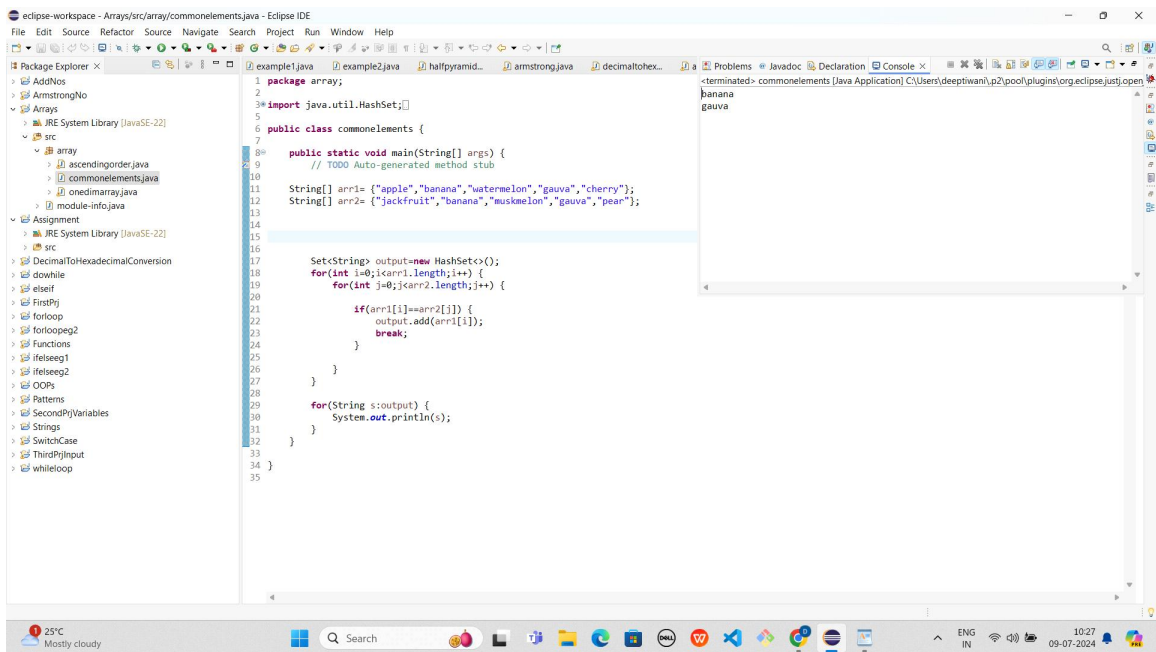
The console on the right shows the output: "Input a decimal no:2545" and "Hexadecimal value is:9F1".

Program to Sort the Elements of an Array in Ascending Order

```
1 package array;
2 import java.util.*;
3 public class ascendingorder {
4
5     // TODO: Auto-generated method stub
6
7     public static void main(String[] args) {
8
9         Scanner sc=new Scanner(System.in);
10
11         int a[]=new int[5];
12         int temp;
13         System.out.print("Enter Elements in array: ");
14         for(int i=0;i<5;i++)
15         {
16             a[i]=sc.nextInt();
17         }
18
19         for(int i=0;i<5;i++) {
20             for(int j=i+1;j<5;j++) {
21                 if(a[i]>a[j]) {
22                     temp=a[i];
23                     a[i]=a[j];
24                     a[j]=temp;
25                 }
26             }
27         }
28
29         for(int i=0;i<5;i++)
30         {
31             System.out.print(a[i] + " ");
32         }
33     }
34 }
35 }
```

-terminated> ascendingorder [Java Application] C:\Users\deepitwani\p2\pooof\plugins\org.eclipse.justj.openjdk...
Enter Elements in array: 78 89 45 32 34
32 34 45 78 89

Program to Find Common Elements Between Two Arrays



The screenshot shows the Eclipse IDE interface. The Package Explorer on the left lists the project structure, including the 'array' package and the 'commonelements.java' file. The main editor displays the following Java code:

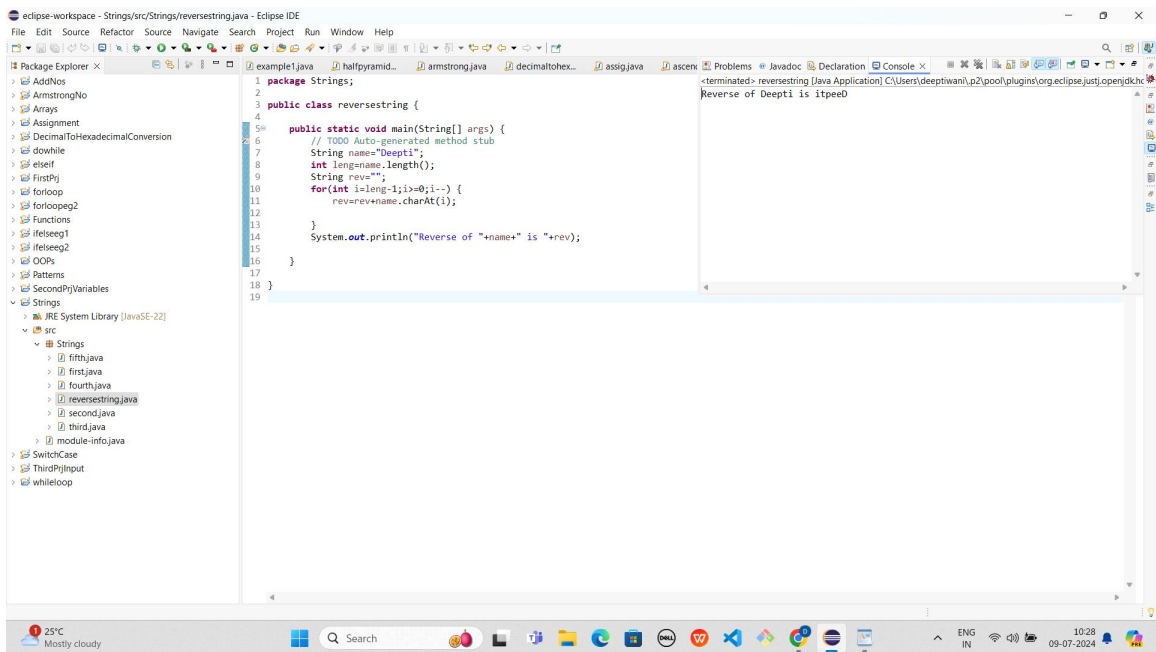
```
1 package array;
2
3 import java.util.HashSet;
4
5 public class commonelements {
6
7
8     public static void main(String[] args) {
9         // TODO Auto-generated method stub
10
11         String[] arr1= {"apple","banana","watermelon","gauva","cherry"};
12         String[] arr2= {"jackfruit","banana","muskmelon","gauva","pear"};
13
14
15
16
17         Set<String> output=new HashSet<>();
18         for(int i=0;i<arr1.length;i++) {
19             for(int j=0;j<arr2.length;j++) {
20
21                 if(arr1[i]==arr2[j]) {
22                     output.add(arr1[i]);
23                     break;
24                 }
25             }
26         }
27
28         for(String s:output) {
29             System.out.println(s);
30         }
31     }
32 }
33
34
35
```

The console on the right shows the output of the program:

```
-terminated> commonelements [Java Application] C:\Users\deepthwan\p2\pool\plugins\org.eclipse.justi.open...
banana
gauva
```

The bottom status bar shows the system clock as 10:27 on 09-07-2024.

Reverse a string in Java

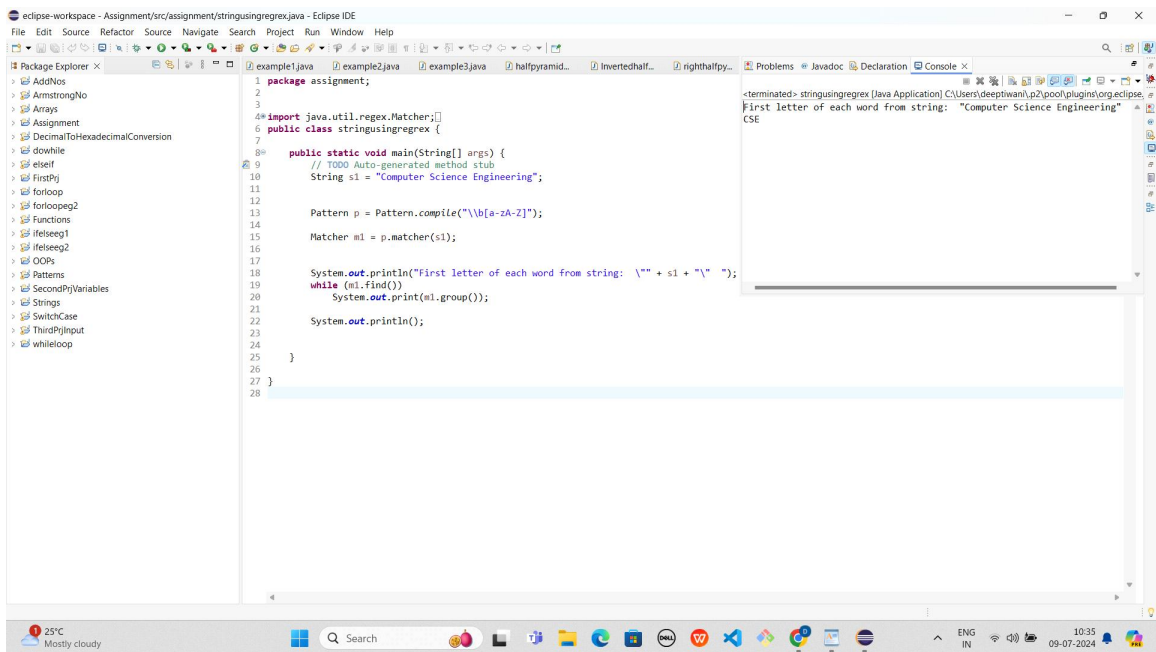


The screenshot shows the Eclipse IDE interface. The Package Explorer on the left lists a project named 'Strings' with several sub-packages and files. The main editor window displays the source code for 'reversestring.java'. The code defines a package 'Strings', a class 'reversestring', and a main method that takes an array of strings as input, reverses each string, and prints the result. The console on the right shows the output of the program: 'Reverse of Deepti is itped'.

```
1 package Strings;
2
3 public class reversestring {
4
5     // TODO: auto-generated method stub
6     public static void main(String[] args) {
7         String name="Deepti";
8         int lengname.length();
9         String rev="";
10        for(int i=leng-1;i>=0;i--) {
11            rev=rev+name.charAt(i);
12        }
13        System.out.println("Reverse of "+name+" is "+rev);
14    }
15 }
16
17
18
19
```

Console Output: -terminated- reversestring [Java Application] C:\Users\deepitiwani\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full\jre\bin\java.exe
Reverse of Deepti is itped

Print first letter of each word in a string using regex



The screenshot shows the Eclipse IDE interface. The Package Explorer on the left lists various Java projects. The main editor displays a Java file named `stringusingregex.java` with the following code:

```
1 package assignment;
2
3
4 import java.util.regex.Matcher;
5
6 public class stringusingregex {
7
8     public static void main(String[] args) {
9         // TODO Auto-generated method stub
10        String s1 = "Computer Science Engineering";
11
12
13        Pattern p = Pattern.compile("\\b[a-zA-Z]");
14
15        Matcher m1 = p.matcher(s1);
16
17
18        System.out.println("First letter of each word from string: '" + s1 + "'");
19        while (m1.find())
20            System.out.print(m1.group());
21
22        System.out.println();
23
24    }
25 }
26
27
28 }
```

The Console window on the right shows the output of the program:

```
<terminated> stringusingregex [Java Application] C:\Users\deepthi\p2\workspace\plugins\org.eclipse
First letter of each word from string: "Computer Science Engineering"
CSE
```

The bottom status bar shows the system temperature as 25°C, mostly cloudy, and the date as 09-07-2024.