Priyansh Salian
C31 - 2003148

Experiment 2: Programs on Basic programming constructs like branching and looping

Theory : Java programming language provides following types of decision making or branching statements in Java . Java programming language provided decision making statements or branching statements as follows .

If statement

An if statement consists of a boolean expression followed by one or more statements .

```
 for (int; testExpression; update){
  //Code
  if(condition to break){
   break;
        }
   }
```

if...else statement

An if statement can be followed by an optional else statement , which executes when the boolean expression is false .

```
if(condition){
commands}

else{

commands

}
```

nested if statement

You can use one if or else if statement inside another if or else if statements .

```
if ( condition )      // Outer if
  if (condition)  // Inner if
    commands ;
```

switch statement

A switch statement allows a variable to be tested for equality against a list of values.

```
switch(expression) {
  case x:
    // code block
    break;
  case y:
    // code block
    break;
  default:
    // code block
}
```

LOOPS

We can initialize the variable, check condition and increment/decrement value. It consists of four parts:

1. **Initialization**: It is the initial condition which is executed once when the loop starts. Here, we can initialize the variable, or we can use an already initialized variable. It is an optional condition.
2. **Condition**: It is the second condition which is executed each time to test the condition of the loop. It continues execution until the condition is false. It must return boolean value either true or false. It is an optional condition.
3. **Increment/Decrement**: It increments or decrements the variable value. It is an optional condition.
4. **Statement**: The statement of the loop is executed each time until the second condition is false.

for loop: The for statement consumes the initialization, condition and increment/decrement in one line thereby providing a shorter, easy to debug structure of looping.

```
for(initialization; condition; increment/decrement){
//statement or code to be executed
}
```

While loop:

Java while loop is a control flow statement that allows code to be executed repeatedly based on a given Boolean condition. The while loop can be thought of as a repeating if statement.

```
while (test_expression)
{
   // statements

  update_expression;
}
```

Do While loop:

The do/while loop is a variant of the while loop. This loop will execute the code block once, before checking if the condition is true, then it will repeat the loop as long as the condition is true.

```
do {
  // code block to be executed
}
while (condition);
```

Java provides three branching statements break, continue and return. The break and continue in Java are two essential keyword beginners needs to familiar while using loops (for loop, while loop and do while loop). break statement in java is used to break the loop and transfers control to the line immediate outside of loop while continue is used to escape current execution (iteration) and transfers control back to the start of the loop. Both break and continue allow the programmer to create sophisticated algorithm and looping constructs.

A.

Aim : WAP to check if a character is a vowel or not .

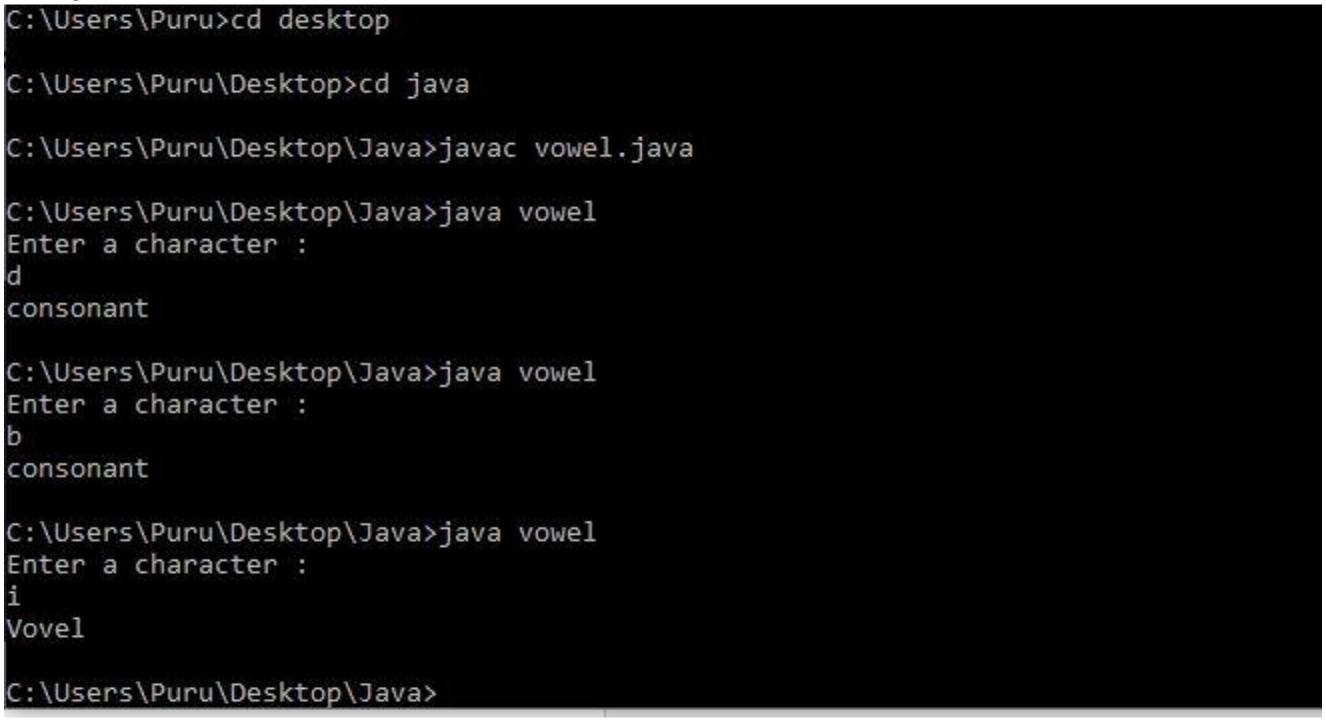
Program :

```
 import java.io.*;

   public class main
 {
       public static void main(String[] args) throws IOException
       {
          BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
          System.out.println("Enter a character :");
          char c = Character.toLowerCase((char)br.read());
          if(c=='a'||c=='e'||c=='i'||c=='o'||c=='u') System.out.println("Vovel");
          else System.out.println("consonant");
       }
   }
```

Puru is my Dad's name so in users his name is appearing

Output :



```
C:\Users\Puru>cd desktop

C:\Users\Puru\Desktop>cd java

C:\Users\Puru\Desktop\Java>javac vowel.java

C:\Users\Puru\Desktop\Java>java vowel
Enter a character :
d
consonant

C:\Users\Puru\Desktop\Java>java vowel
Enter a character :
b
consonant

C:\Users\Puru\Desktop\Java>java vowel
Enter a character :
i
Vovel

C:\Users\Puru\Desktop\Java>
```

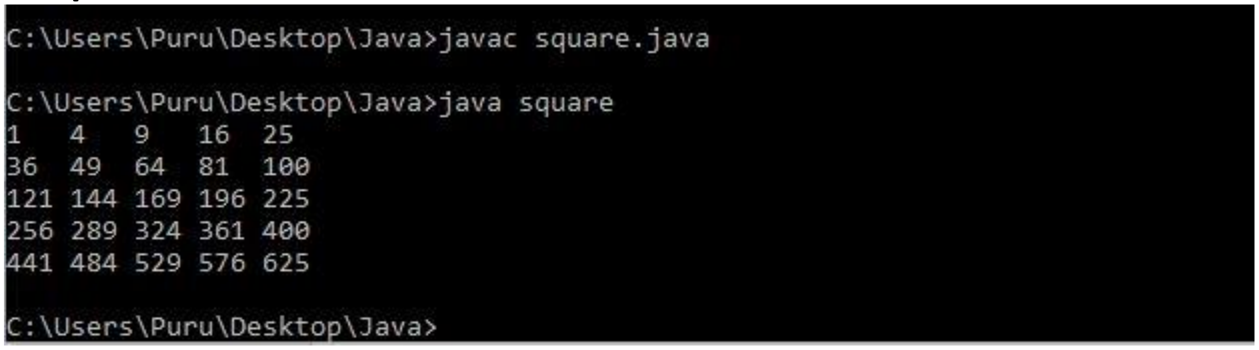
B.

Aim : WAP to print a two dimensional table of squares of numbers from 1 to 25 using for loop .

Program :
import java.io.*;

```
 public class squares {
    public static void main(String[] args) {
       int i;
       for(i=1;i<=25;i++)
       {
          if(i*i<10) System.out.print(i*i+"   ");
          else if(i*i<100) System.out.print(i*i+"  ");
          else System.out.print(i*i+" ");
          if(i%5==0) System.out.println("");
       }
    }
 }
```

Output :



```
C:\Users\Puru\Desktop\Java>javac square.java

C:\Users\Puru\Desktop\Java>java square
1   4   9   16  25
36  49  64  81  100
121 144 169 196 225
256 289 324 361 400
441 484 529 576 625

C:\Users\Puru\Desktop\Java>
```

C.

Aim : WAP to find number of and sum of all integers greater than 100 and less than 200 that are divisible by 7 .

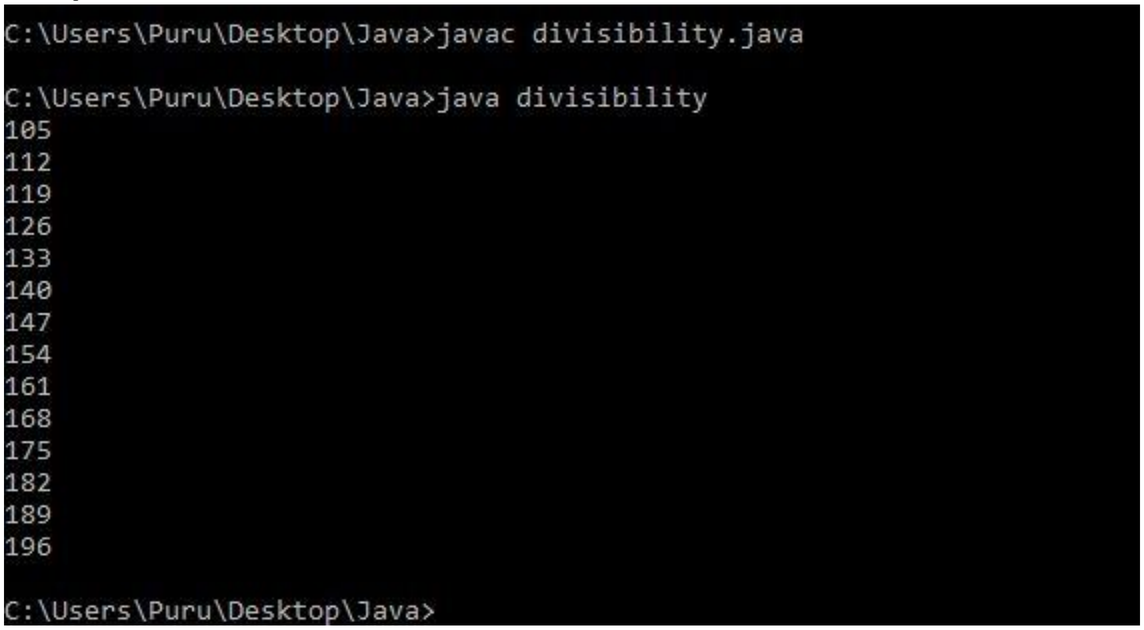
Program :

```
import java.io.*;


public class divisibility {
   public static void main(String[] args) {

      for(int i=100;i<=200;i++)
      {
         if(i%7==0) System.out.println(i);
      }
   }
}
```

Output :



```
C:\Users\Puru\Desktop\Java>javac divisibility.java

C:\Users\Puru\Desktop\Java>java divisibility
105
112
119
126
133
140
147
154
161
168
175
182
189
196

C:\Users\Puru\Desktop\Java>
```

D.

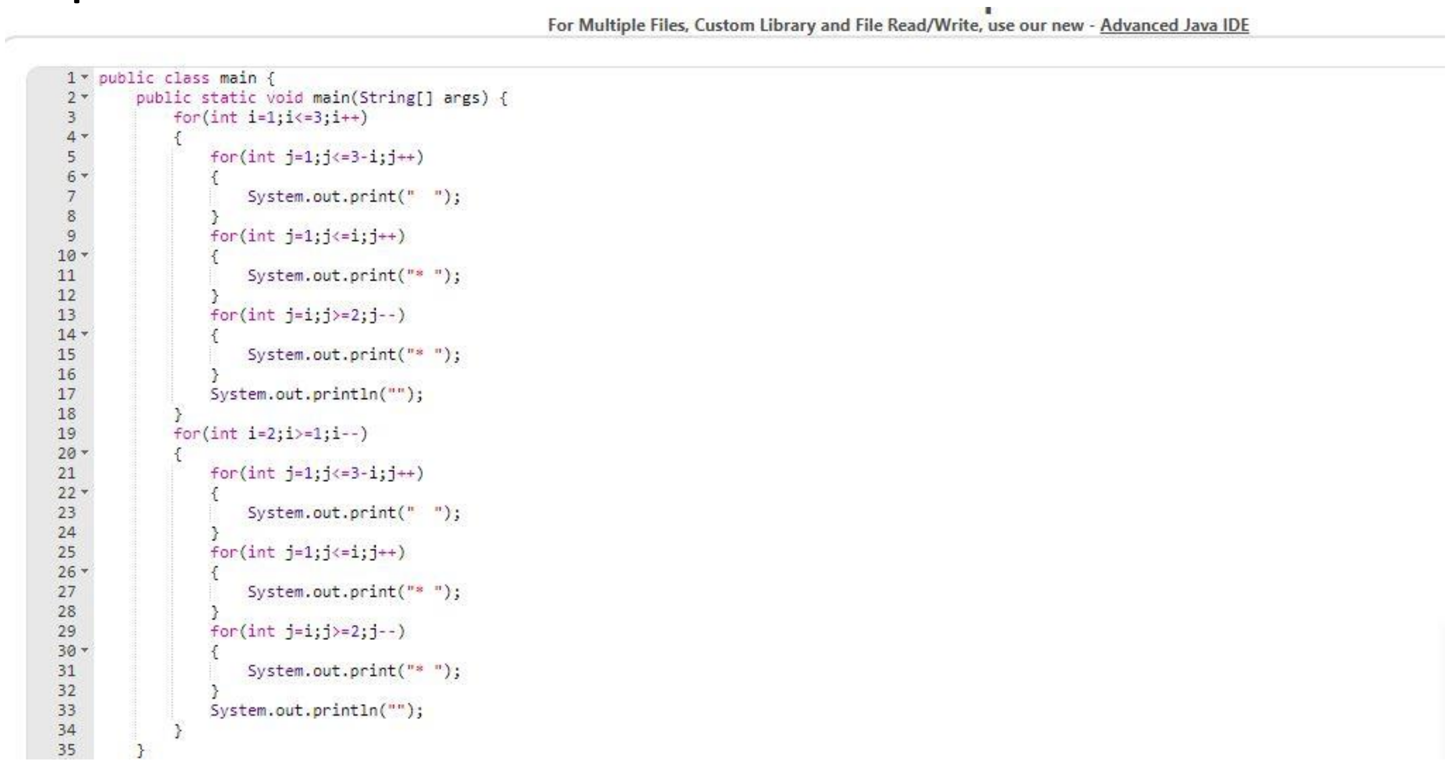
Aim : WAP to print the following pattern,

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

Program :

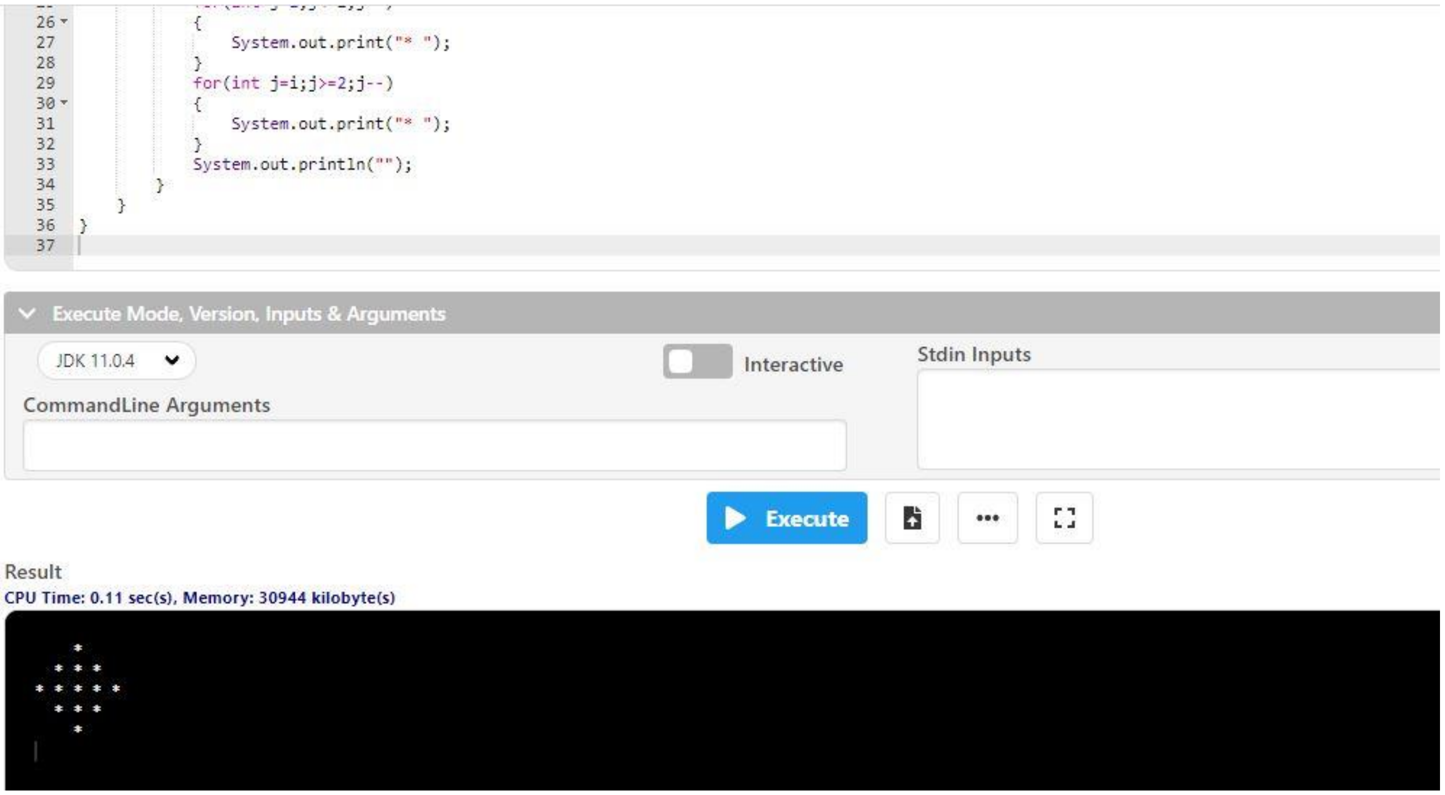
```
public class main {
   public static void main(String[] args) {
      for(int i=1;i<=3;i++)
      {
         for(int j=1;j<=3-i;j++)
         {
            System.out.print("  ");
         }
         for(int j=1;j<=i;j++)
         {
            System.out.print("* ");
         }
         for(int j=i;j>=2;j--)
         {
            System.out.print("* ");
         }
         System.out.println("");
      }
      for(int i=2;i>=1;i--)
      {
         for(int j=1;j<=3-i;j++)
         {
            System.out.print("  ");
         }
         for(int j=1;j<=i;j++)
         {
          System.out.print("* ");
        }
        for(int j=i;j>=2;j--)
        {
          System.out.print("* ");
        }
        System.out.println("");
      }
    }
}
```

Output :



For Multiple Files, Custom Library and File Read/Write, use our new - Advanced Java IDE

```
public class main {
    public static void main(String[] args) {
        for(int i=1;i<=3;i++)
        {
            for(int j=1;j<=3-i;j++)
            {
                System.out.print("  ");
            }
            for(int j=1;j<=i;j++)
            {
                System.out.print("* ");
            }
            for(int j=i;j>=2;j--)
            {
                System.out.print("* ");
            }
            System.out.println("");
        }
        for(int i=2;i>=1;i--)
        {
            for(int j=1;j<=3-i;j++)
            {
                System.out.print("  ");
            }
            for(int j=1;j<=i;j++)
            {
                System.out.print("* ");
            }
            for(int j=i;j>=2;j--)
            {
                System.out.print("* ");
            }
            System.out.println("");
        }
    }
```



```
            {
                System.out.print("* ");
            }
            for(int j=i;j>=2;j--)
            {
                System.out.print("* ");
            }
            System.out.println("");
        }
    }
}
```

Result

CPU Time: 0.11 sec(s), Memory: 30944 kilobyte(s)

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