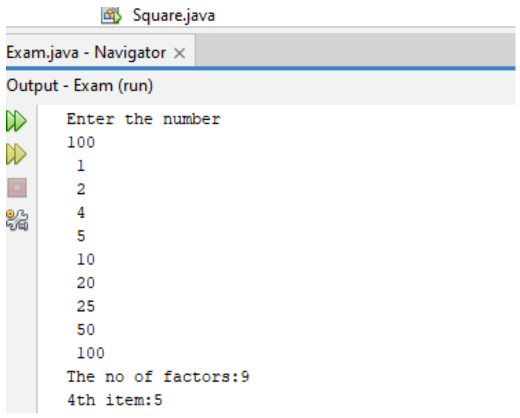
1. Question for Finding error in Java to determine the factor

Program:

```
import java.util.*;

public class Exam {

    public static void main(String[] args) {

        // TODO code application logic here

        int count=0,n,i,j=0,m=4;

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter the number");

        n = sc.nextInt();

        int a[] = new int[10];

       try

       {

          if(n<=0)

          {

             System.out.println("Enter valid number");

          }

          else

          {

            for(i=1;i<=n;i++)

            {

if(n%i==0)

{

a[j] = i;
System.out.println(" "+ i);

count++;

j++;

}

}

System.out.println("The no of factors:"+count);

      }

      System.out.println(m +"th item:"+ a[m-1]);

     }

    catch (Exception e)

    {

      System.out.println("Enter only numbers");

    }

  }

}
```

Output:



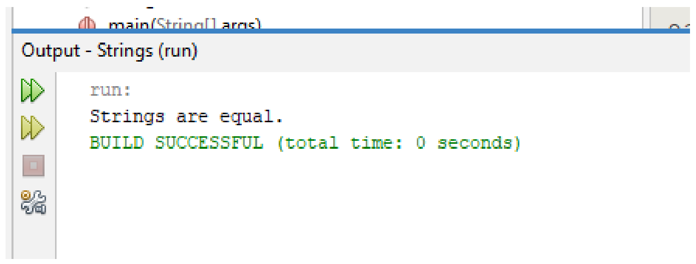
```
Enter the number
100
 1
 2
 4
 5
 10
 20
 25
 50
 100
The no of factors:9
4th item:5
```

1. **Write a java program**
   i. to compare two strings lexicographically, ignoring case differences.

   Program:

```
public class Strings {

  public static void main(String[] args)

      String str1 = "Hello";

   String str2 = "hello";

   int result = str1.compareToIgnoreCase(str2);

   if (result == 0) {

     System.out.println("Strings are equal.");

   }

   else if (result < 0) {

     System.out.println("String 1 comes before string 2.");

   }

   else {

     System.out.println("String 1 comes after string 2.");

   }

 }

}
```

Output:



```
run:
Strings are equal.
BUILD SUCCESSFUL (total time: 0 seconds)
```

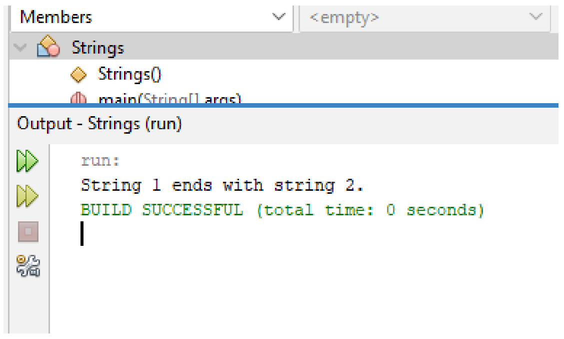
ii.   to check whether a given string ends with the contents of another string.

Program:

```
String str1 = "Hello World";

   String str2 = "World";
```

```
    boolean result = str1.endsWith(str2);

    if (result) {

      System.out.println("String 1 ends with string 2.");

    }

    else {

      System.out.println("String 1 does not end with string 2.");

    }

  }

}
```

Output:



```
run:
String 1 ends with string 2.
BUILD SUCCESSFUL (total time: 0 seconds)
```

iii. to print current date and time in the specified format.

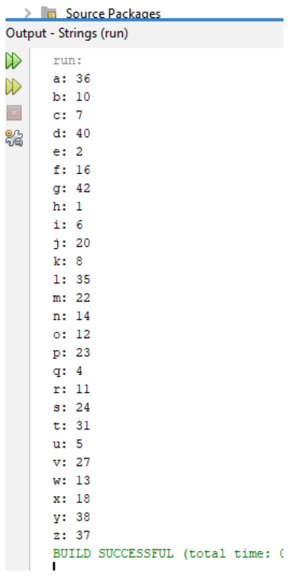
Program:

iv.to get the index of all the characters of the alphabet.

Program:

```
String str = "The quick brown fox jumps over the lazy dog.";

    for (char ch = 'a'; ch <= 'z'; ch++) {

      int index = str.indexOf(ch);

      if (index != -1) {

        System.out.println(ch + ": " + index);
        }
      }
   }
}
```

Output:



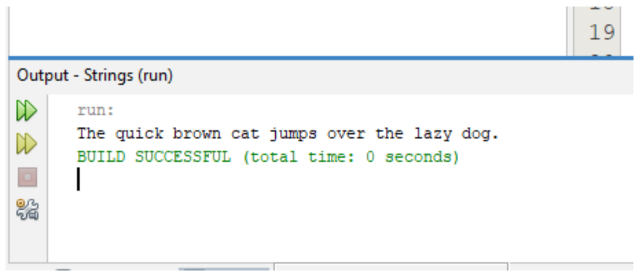
```
run:
a: 36
b: 10
c: 7
d: 40
e: 2
f: 16
g: 42
h: 1
i: 6
j: 20
k: 8
l: 35
m: 22
n: 14
o: 12
p: 23
q: 4
r: 11
s: 24
t: 31
u: 5
v: 27
w: 13
x: 18
y: 38
z: 37
BUILD SUCCESSFUL (total time: (
```

v. To replace each substring of a given string that matches the given regular expression with the given replacement. In the below string replace all the fox with cat.

Program:

```
public class Strings {
    public static void main(String[] args) {
        // TODO code application logic here
            String str = "The quick brown fox jumps over the lazy dog.";
      String newStr = str.replaceAll("fox", "cat");
      System.out.println(newStr);
   }
}
```

Output:

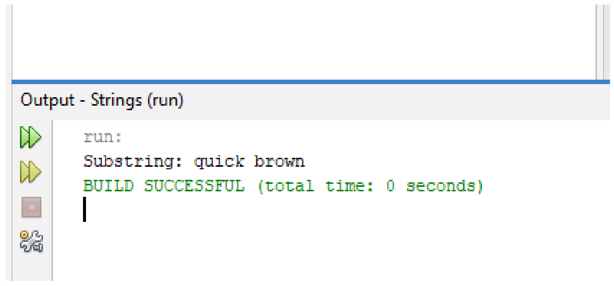

```
run:
The quick brown cat jumps over the lazy dog.
BUILD SUCCESSFUL (total time: 0 seconds)
```

vi. to get a substring of a given string between two specified positions.

Program:

```
public class Strings {

   public static void main(String[] args) {

    String str = "The quick brown fox jumps over the lazy dog.";

     String substr = str.substring(4, 15);

     System.out.println("Substring: " + substr);

  }

}
```

Output:


```
run:
Substring: quick brown
BUILD SUCCESSFUL (total time: 0 seconds)
```

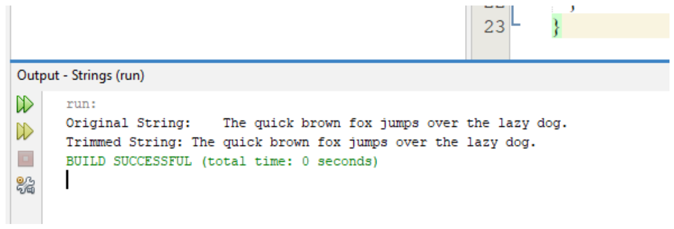
vii. to trim any leading or trailing whitespace from a given string.

Program:

```
public class Strings {

   public static void main(String[] args) {

       // TODO code application logic here

     String str = "   The quick brown fox jumps over the lazy dog.   ";
```

```
      String trimmedStr = str.trim();

      System.out.println("Original String: " + str);

      System.out.println("Trimmed String: " + trimmedStr);

  }

}
```

Output:



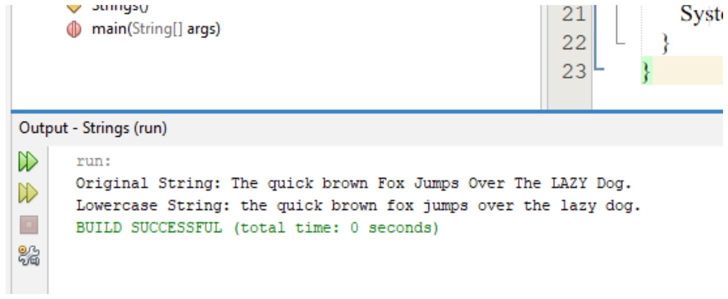
```
run:
Original String:    The quick brown fox jumps over the lazy dog.
Trimmed String: The quick brown fox jumps over the lazy dog.
BUILD SUCCESSFUL (total time: 0 seconds)
```

viii. to convert all the characters in a string to lowercase.

Program:

```
public class Strings {

    public static void main(String[] args) {

       String str = "The quick brown Fox Jumps Over The LAZY Dog.";

      String lowercaseStr = str.toLowerCase();

      System.out.println("Original String: " + str);

      System.out.println("Lowercase String: " + lowercaseStr);

  }

}
```

Output:



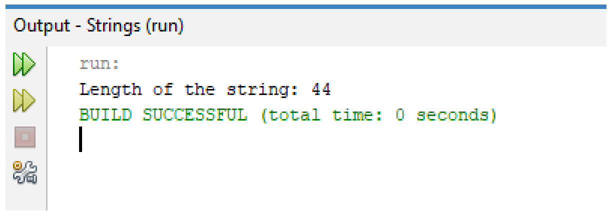
```
run:
Original String: The quick brown Fox Jumps Over The LAZY Dog.
Lowercase String: the quick brown fox jumps over the lazy dog.
BUILD SUCCESSFUL (total time: 0 seconds)
```

ix. to get the length of a given string.

Program:

```
public class Strings {

   public static void main(String[] args) {

     String str = "The quick brown fox jumps over the lazy dog.";

     int len = str.length();

     System.out.println("Length of the string: " + len);

  }

}
```

Output:


Output - Strings (run)

```
run:
Length of the string: 44
BUILD SUCCESSFUL (total time: 0 seconds)
```

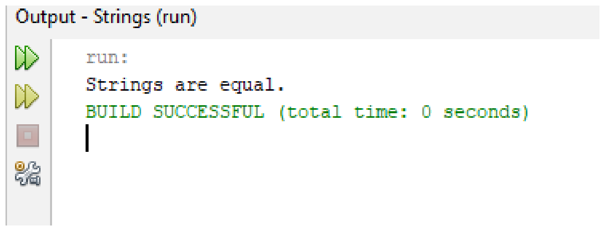
x. to check whether two String objects contain the same data

Program:

```
String str1 = "The quick brown fox jumps over the lazy dog.";

     String str2 = "The quick brown fox jumps over the lazy dog.";

     if (str1.equals(str2)) {

       System.out.println("Strings are equal.");

     }

     else {

       System.out.println("Strings are not equal.");

     }

  }
```

}

Output:



Output - Strings (run)

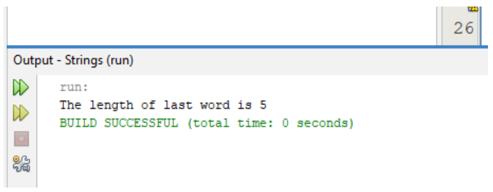
```
run:
Strings are equal.
BUILD SUCCESSFUL (total time: 0 seconds)
```

2. Given a string s consisting of words and spaces, return the length of the last word in the string.

Program:

```
public class Strings {
  public static int lengthOfLastWord(String s) {
  s = s.trim();
  int lastSpaceIndex = s.lastIndexOf(' ');
  if (lastSpaceIndex == -1) {
    return s.length();
  } else {
    return s.substring(lastSpaceIndex + 1).length();
  }
}
  public static void main(String[] args)
    String input = "Hello World";
    Strings a = new Strings();
    System.out.println("The length of last word is "+a.lengthOfLastWord(input));
  }
}
```

Output:



```
run:
The length of last word is 5
BUILD SUCCESSFUL (total time: 0 seconds)
```

2. Implement a class Account. An account has

 a balance

 functions to add

 and withdraw money,

 And a function to inquire the current balance.

Condition:

1. Pass a value into a constructor to set an initial balance.

2. If no value is passed the initial balance should be set to $0.

3. Charge a $5 penalty if an attempt is made to withdraw more money than available

in the account.

Enhance the Account class to compute interest on the current balance.

Program:

```
import java.util.*;
public class account {
    private double balance;
    private double interestRate;
    private static final double PENALTY = 5.0;

    public account() {
      this(0.0);
    }

    public account(double initialBalance) {
      balance = initialBalance;
      interestRate = 0.0;
    }

    public void deposit(double amount) {
      balance += amount;
    }

    public void withdraw(double amount) {
      if (balance >= amount) {
        balance -= amount;
      } else {
        balance -= PENALTY;
        System.out.println("Withdrawal amount exceeds account balance. A $5 penalty has
been charged.");
      }
    }

    public double getBalance() {
      return balance;
    }

    public void setInterestRate(double rate) {
      interestRate = rate;
    }

    public void addInterest() {
      double interest = balance * interestRate / 100.0;
      balance += interest;
    }
  }
```

3. Given two strings needle and haystack, return the index of the first occurrence of

needle in haystack, or -1 if needle is not part of haystack.

Program:

```
public class firstoccurance {

    public int strStr(String haystack, String needle) {

      if (needle.isEmpty()) {

        return 0;

      }

      int needleLength = needle.length();

      for (int i = 0; i <= haystack.length() - needleLength; i++) {
            if (haystack.substring(i, i + needleLength).equals(needle)) {
                return i;
            }
        }
        return -1;
    }
}
```

Output: