Text Processing

Mitsu Ogihara

Department of Computer Science
University of Miami

Table of Contents

Class char

Class char is a primitive data type demanding 16 bits Each char type can be treated as a number by either

- putting `(int)` before it (that is, **casting**) or
- putting in a mathematical formula

```
char c = 'a';
int value = (int)c;
int result = c - 10;
```

A positive integer can be turned into a char by putting `(char)` before it (that is, **casting**)

```
char c = 'a';
char d = (char)( c + 1 );
```

Char vs. Int

Receive from the user an input line and for each position of the input string, print the char, the int value, and the next char

Char vs. Int

```
import java.util.*;
public class PlayWithChar {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a string and hit return: " );
    String input = console.nextLine();
    System.out.println( "pos\tchar\tint\tnext" );
    for ( int pos = 0; pos < input.length(); pos ++ ) {
      char ch = input.charAt( pos );
      char shifted = (char)( ch + 1 );
      int value = (int)ch;
      System.out.println( pos + "\t" + ch
          + "\t" + value + "\t" + shifted );
    }
  }
}
```

Scanner creation

Char vs. Int

```
import java.util.*;
public class PlayWithChar {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a string and hit return: " );
    String input = console.nextLine();
    System.out.println( "pos\tchar\tint\tnext" );
    for ( int pos = 0; pos < input.length(); pos ++ ) {
      char ch = input.charAt( pos );
      char shifted = (char)( ch + 1 );
      int value = (int)ch;
      System.out.println( pos + "\t" + ch
          + "\t" + value + "\t" + shifted );
    }
  }
}
```

Prompt the user to receive input

Char vs. Int

```java
import java.util.*;
public class PlayWithChar {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a string and hit return: " );
    String input = console.nextLine();
    System.out.println( "pos\tchar\tint\tnext" );
    for ( int pos = 0; pos < input.length(); pos ++ ) {
      char ch = input.charAt( pos );
      char shifted = (char)( ch + 1 );
      int value = (int)ch;
      System.out.println( pos + "\t" + ch
          + "\t" + value + "\t" + shifted );
    }
  }
}
```

Print the header line

Char vs. Int

```java
import java.util.*;
public class PlayWithChar {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a string and hit return: " );
    String input = console.nextLine();
    System.out.println( "pos\tchar\tint\tnext" );
    for ( int pos = 0; pos < input.length(); pos ++ ) {
      char ch = input.charAt( pos );
      char shifted = (char)( ch + 1 );
      int value = (int)ch;
      System.out.println( pos + "\t" + ch
          + "\t" + value + "\t" + shifted );
    }
  }
}
```

For loop

Char vs. Int

```java
import java.util.*;
public class PlayWithChar {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a string and hit return: " );
    String input = console.nextLine();
    System.out.println( "pos\tchar\tint\tnext" );
    for ( int pos = 0; pos < input.length(); pos ++ ) {
      char ch = input.charAt( pos );
      char shifted = (char)( ch + 1 );
      int value = (int)ch;
      System.out.println( pos + "\t" + ch
          + "\t" + value + "\t" + shifted );
    }
  }
}
```

Obtain the character at the position, the next character, and the integer value of the character

Char vs. Int

```
import java.util.*;
public class PlayWithChar {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a string and hit return: " );
    String input = console.nextLine();
    System.out.println( "pos\tchar\tint\tnext" );
    for ( int pos = 0; pos < input.length(); pos ++ ) {
      char ch = input.charAt( pos );
      char shifted = (char)( ch + 1 );
      int value = (int)ch;
      System.out.println( pos + "\t" + ch
          + "\t" + value + "\t" + shifted );
    }
  }
}
```

Print the information

Class Character

Class Character offers a number of methods for processing information stored in a char

- `Character.getNumericValue( char ch )`: returns a nonnegative number if the char represents a number (0-9 for '0'-'9'; 10-35 for 'a'-'z' and 'A'-'Z'); a negative value otherwise
- `Character.isDigit( char ch )`: returns in boolean whether the char is a digit
- `Character.isLetter( char ch )`: returns in boolean whether the char is a letter of the alphabet
- `Character.isUpperCase( char ch )`: returns in boolean whether the char is an upper case letter
- `Character.isLowerCase( char ch )`: returns in boolean whether the char is a lower case letter
- `Character.toUpperCase( char ch )`: returns the upper case letter of the char
- `Character.toLowerCase( char ch )`: returns the lower case letter of the char

Character Methods

Receive from the user an input line and for each position of the input string, print the result of the above methods

Character Methods

```java
import java.util.*;
public class CharacterMethods {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a string and hit return: " );
    String input = console.nextLine();
    System.out.println( "pos\tchar\tnumVal\tisDig\tisLet"
        + "\tisLo\tisUp\ttoLo\ttoUp" );
    for ( int pos = 0; pos < input.length(); pos ++ ) {
      char ch = input.charAt( pos );
      System.out.println( pos + "\t" + ch
          + "\t" +  Character.getNumericValue( ch )
          + "\t" +  Character.isDigit( ch )
          + "\t" +  Character.isLetter( ch )
          + "\t" +  Character.isLowerCase( ch )
          + "\t" +  Character.isUpperCase( ch )
          + "\t" +  Character.toLowerCase( ch )
          + "\t" +  Character.toUpperCase( ch ) );
    }
  }
}
```

Scanner creation

Character Methods

```java
import java.util.*;
public class CharacterMethods {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a string and hit return: " );
    String input = console.nextLine();
    System.out.println( "pos\tchar\tnumVal\tisDig\tisLet"
        + "\tisLo\tisUp\ttoLo\ttoUp" );
    for ( int pos = 0; pos < input.length(); pos ++ ) {
      char ch = input.charAt( pos );
      System.out.println( pos + "\t" + ch
         + "\t" +  Character.getNumericValue( ch )
         + "\t" +  Character.isDigit( ch )
         + "\t" +  Character.isLetter( ch )
         + "\t" +  Character.isLowerCase( ch )
         + "\t" +  Character.isUpperCase( ch )
         + "\t" +  Character.toLowerCase( ch )
         + "\t" +  Character.toUpperCase( ch ) );
    }
  }
}
```

Prompt the user to receive input

Character Methods

```
import java.util.*;
public class CharacterMethods {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a string and hit return: " );
    String input = console.nextLine();
    System.out.println( "pos\tchar\tnumVal\tisDig\tisLet"
        + "\tisLo\tisUp\ttoLo\ttoUp" );
    for ( int pos = 0; pos < input.length(); pos ++ ) {
      char ch = input.charAt( pos );
      System.out.println( pos + "\t" + ch
         + "\t" +  Character.getNumericValue( ch )
         + "\t" +  Character.isDigit( ch )
         + "\t" +  Character.isLetter( ch )
         + "\t" +  Character.isLowerCase( ch )
         + "\t" +  Character.isUpperCase( ch )
         + "\t" +  Character.toLowerCase( ch )
         + "\t" +  Character.toUpperCase( ch ) );
    }
  }
}
```

Print the header line

Character Methods

```java
import java.util.*;
public class CharacterMethods {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a string and hit return: " );
    String input = console.nextLine();
    System.out.println( "pos\tchar\tnumVal\tisDig\tisLet"
        + "\tisLo\tisUp\ttoLo\ttoUp" );
    for ( int pos = 0; pos < input.length(); pos ++ ) {
      char ch = input.charAt( pos );
      System.out.println( pos + "\t" + ch
         + "\t" +  Character.getNumericValue( ch )
         + "\t" +  Character.isDigit( ch )
         + "\t" +  Character.isLetter( ch )
         + "\t" +  Character.isLowerCase( ch )
         + "\t" +  Character.isUpperCase( ch )
         + "\t" +  Character.toLowerCase( ch )
         + "\t" +  Character.toUpperCase( ch ) );
    }
  }
}
```

For loop

Character Methods

```java
import java.util.*;
public class CharacterMethods {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a string and hit return: " );
    String input = console.nextLine();
    System.out.println( "pos\tchar\tnumVal\tisDig\tisLet"
        + "\tisLo\tisUp\ttoLo\ttoUp" );
    for ( int pos = 0; pos < input.length(); pos ++ ) {
      char ch = input.charAt( pos );
      System.out.println( pos + "\t" + ch
         + "\t" +  Character.getNumericValue( ch )
         + "\t" +  Character.isDigit( ch )
         + "\t" +  Character.isLetter( ch )
         + "\t" +  Character.isLowerCase( ch )
         + "\t" +  Character.isUpperCase( ch )
         + "\t" +  Character.toLowerCase( ch )
         + "\t" +  Character.toUpperCase( ch ) );
    }
  }
}
```

Obtain the character at the position

Character Methods

```java
import java.util.*;
public class CharacterMethods {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a string and hit return: " );
    String input = console.nextLine();
    System.out.println( "pos\tchar\tnumVal\tisDig\tisLet"
        + "\tisLo\tisUp\ttoLo\ttoUp" );
    for ( int pos = 0; pos < input.length(); pos ++ ) {
      char ch = input.charAt( pos );
      System.out.println( pos + "\t" + ch
         + "\t" +  Character.getNumericValue( ch )
         + "\t" +  Character.isDigit( ch )
         + "\t" +  Character.isLetter( ch )
         + "\t" +  Character.isLowerCase( ch )
         + "\t" +  Character.isUpperCase( ch )
         + "\t" +  Character.toLowerCase( ch )
         + "\t" +  Character.toUpperCase( ch ) );
    }
  }
}
```

Print the information

Table of Contents

Printf for Formatted Print

`System.out.printf` is a method that provides formatted printing
The syntax is

```
System.out.printf(<parameters>, <data_1>, ..., <data_k>)
```

where <paramters> is a String with k format parameters embedded

- Each parameter starts with a '%' and ends with some special letter or letters, which specify that the data items are String, a real number, and an integer, respectively
- These specified data types need to match with the k data items
- We study only 's', 'f', and 'd' for the special letter

printf Specification Examples

- `%11s`: for a String; use at least 11 character spaces
- `%-11.5s`: for a String; use 11 character spaces and print no more than 5 letters of the String; do this left-aligned
- `%-12d`: for an integer; use at least 12 character spaces and left-aligned
- `%+12d`: for an integer; use at least 12 character spaces and produce ’+’ for a positive number
- `%+012d`: for an integer; use at least 12 character spaces and produce ’+’ for a positive number with leading 0’s
- `%-13.3f`: for an integer; allocate exactly 13 character spaces with exactly 3 digits below the decimal point; do this left-aligned

printf Specification Examples

- `%11s`: for a String; use at least 11 character spaces
- `%-11.5s`: for a String; use 11 character spaces and print no more than 5 letters of the String; do this left-aligned
- `%-12d`: for an integer; use at least 12 character spaces and left-aligned
- `%+12d`: for an integer; use at least 12 character spaces and produce '+' for a positive number
- `%+012d`: for an integer; use at least 12 character spaces and produce '+' for a positive number with leading 0's
- `%-13.3f`: for an integer; allocate exactly 13 character spaces with exactly 3 digits below the decimal point; do this left-aligned

Non-specification strings can be added before, after, or between specifications, e.g., `"abc=%03d and def=%10.1f!"`, where "abc=", " and def=", and "!" are non-specifications

A Simple Example

Receive from the user a positive integer n, compute sin, cos, tan of $(\frac{i}{n})\pi$ for $i = 0, ..., 2n$ (that is, the full circle), and print the values along with the angle values in degree and in radian

Trigonometry Calculation

```
import java.util.*;
public class TrigonometryCalculation {
 public static void main( String[] args ) {
  Scanner console = new Scanner( System.in );
  System.out.print( "Enter a positive n:  " );
  int n = console.nextInt();
  if ( n <= 0 ) {
   throw new IllegalArgumentException(
     "number must be positive:  " + n );
  }
  for ( int i = 0; i <= 2 * n; i ++ ) {
   double deg = 180.0 * i / n;
   double rad = (double)i / n;
   double sin = Math.sin( rad );
   double cos = Math.cos( rad );
   double tan = Math.tan( rad );
   System.out.printf(
     "i=%3d, deg=%6.2f, rad=%6.2f, sin=%6.2f, "
     + "cos=%6.2f, tan=%6.2f%n",
     i, deg, rad, sin, cos, tan );
  }
 }
}
```

Scanner creation, prompt the user to receive input, and error handling

Trigonometry Calculation

```java
import java.util.*;
public class TrigonometryCalculation {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a positive n:  " );
    int n = console.nextInt();
    if ( n <= 0 ) {
      throw new IllegalArgumentException(
          "number must be positive:  " + n );
    }
    for ( int i = 0; i <= 2 * n; i ++ ) {
      double deg = 180.0 * i / n;
      double rad = (double)i / n;
      double sin = Math.sin( rad );
      double cos = Math.cos( rad );
      double tan = Math.tan( rad );
      System.out.printf(
          "i=%3d, deg=%6.2f, rad=%6.2f, sin=%6.2f, "
          + "cos=%6.2f, tan=%6.2f%n",
          i, deg, rad, sin, cos, tan );
    }
  }
}
```

For loop; note the termination condition is `i <= 2 * n`

Trigonometry Calculation

```
import java.util.*;
public class TrigonometryCalculation {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a positive n: " );
    int n = console.nextInt();
    if ( n <= 0 ) {
      throw new IllegalArgumentException(
          "number must be positive: " + n );
    }
    for ( int i = 0; i <= 2 * n; i ++ ) {
      double deg = 180.0 * i / n;
      double rad = (double)i / n;
      double sin = Math.sin( rad );
      double cos = Math.cos( rad );
      double tan = Math.tan( rad );
      System.out.printf(
          "i=%3d, deg=%6.2f, rad=%6.2f, sin=%6.2f, "
          + "cos=%6.2f, tan=%6.2f%n",
          i, deg, rad, sin, cos, tan );
    }
  }
}
```

Compute the degree as `180.0 * i / n`
the `.0` is important, otherwise the right-hand side will be processed as an integer!

Trigonometry Calculation

```java
import java.util.*;
public class TrigonometryCalculation {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a positive n: " );
    int n = console.nextInt();
    if ( n <= 0 ) {
      throw new IllegalArgumentException(
          "number must be positive: " + n );
    }
    for ( int i = 0; i <= 2 * n; i ++ ) {
      double deg = 180.0 * i / n;
      double rad = (double)i / n;
      double sin = Math.sin( rad );
      double cos = Math.cos( rad );
      double tan = Math.tan( rad );
      System.out.printf(
          "i=%3d, deg=%6.2f, rad=%6.2f, sin=%6.2f, "
          + "cos=%6.2f, tan=%6.2f%n",
          i, deg, rad, sin, cos, tan );
    }
  }
}
```

Compute the radian as `(double)i / n` by way of casting

Trigonometry Calculation

```java
import java.util.*;
public class TrigonometryCalculation {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a positive n: " );
    int n = console.nextInt();
    if ( n <= 0 ) {
      throw new IllegalArgumentException(
          "number must be positive: " + n );
    }
    for ( int i = 0; i <= 2 * n; i ++ ) {
      double deg = 180.0 * i / n;
      double rad = (double)i / n;
      double sin = Math.sin( rad );
      double cos = Math.cos( rad );
      double tan = Math.tan( rad );
      System.out.printf(
          "i=%3d, deg=%6.2f, rad=%6.2f, sin=%6.2f, "
          + "cos=%6.2f, tan=%6.2f%n",
          i, deg, rad, sin, cos, tan );
    }
  }
}
```

Compute the trigonometry function values

Trigonometry Calculation

```java
import java.util.*;
public class TrigonometryCalculation {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter a positive n: " );
    int n = console.nextInt();
    if ( n <= 0 ) {
      throw new IllegalArgumentException(
          "number must be positive: " + n );
    }
    for ( int i = 0; i <= 2 * n; i ++ ) {
      double deg = 180.0 * i / n;
      double rad = (double)i / n;
      double sin = Math.sin( rad );
      double cos = Math.cos( rad );
      double tan = Math.tan( rad );
      System.out.printf(
          "i=%3d, deg=%6.2f, rad=%6.2f, sin=%6.2f, "
          + "cos=%6.2f, tan=%6.2f%n",
          i, deg, rad, sin, cos, tan );
    }
  }
}
```

Print the information; "%n" at the end of the format parameter String means the newline of the machine

Table of Contents

Return

The return statement terminates the execution of the method in which it appears, and so, instead of writing

```
if ( <codition> ) {
  return X;
}
else {
  return Y;
}
```

You may write

```
if ( <codition> ) {
  return X;
}
return Y;
```

Comparing Two Strings

A method for mimicking the `compareTo` method of the class `String`

- Design it as a static method that takes two `String` objects `s` and `t`
- The method compares them character by character, starting from the beginnning
  - Find the first position they differ, if at all, and return -1 if the character of `s` is smaller than the character of `t` and $+1$ otherwise
  - If no such position is found, return -1 if `s` is shorter than `t`, $+1$ if `s` is longer than `t`, and 0 otherwise

Examples

s	t	return value
`abc`	`abdef`	-1
`abc`	`Xb`	$+1$
`abc`	`abc`	0
`abc`	`abcdef`	-1

String Comparison as a Static Method

Receive from the user two strings and perform comparison to match the output requirement of the `compareTo` method of class `String`

StringComparison.java

```java
import java.util.Scanner;
public class StringComparison {
  public static int compare( String s, String t ) {
    int leng = Math.min( s.length(), t.length() );
    for ( int i = 0; i < leng; i ++ ) {
      if ( s.charAt( i ) < t.charAt( i ) ) {
        return -1;
      }
      if ( s.charAt( i ) > t.charAt( i ) ) {
        return +1;
      }
    }
    if ( s.length() < t.length() ) {
      return -1;
    }
    if ( s.length() > t.length() ) {
      return +1;
    }
    return 0;
  }
```

Method for comparison - declaration

StringComparison.java

```java
import java.util.Scanner;
public class StringComparison {
 public static int compare( String s, String t ) {
  int leng = Math.min( s.length(), t.length() );
  for ( int i = 0; i < leng; i ++ ) {
   if ( s.charAt( i ) < t.charAt( i ) ) {
    return -1;
   }
   if ( s.charAt( i ) > t.charAt( i ) ) {
    return +1;
   }
  }
  if ( s.length() < t.length() ) {
   return -1;
  }
  if ( s.length() > t.length() ) {
   return +1;
  }
  return 0;
 }
```

Determine the length of the shorter of the two strings

StringComparison.java

```java
import java.util.Scanner;
public class StringComparison {
  public static int compare( String s, String t ) {
    int leng = Math.min( s.length(), t.length() );
    for ( int i = 0; i < leng; i ++ ) {
      if ( s.charAt( i ) < t.charAt( i ) ) {
        return -1;
      }
      if ( s.charAt( i ) > t.charAt( i ) ) {
        return +1;
      }
    }
    if ( s.length() < t.length() ) {
      return -1;
    }
    if ( s.length() > t.length() ) {
      return +1;
    }
    return 0;
  }
```

For loop that goes through position indices from 0 to the length of the shorter one -1

StringComparison.java

```java
import java.util.Scanner;
public class StringComparison {
  public static int compare( String s, String t ) {
    int leng = Math.min( s.length(), t.length() );
    for ( int i = 0; i < leng; i ++ ) {
      if ( s.charAt( i ) < t.charAt( i ) ) {
        return -1;
      }
      if ( s.charAt( i ) > t.charAt( i ) ) {
        return +1;
      }
    }
    if ( s.length() < t.length() ) {
      return -1;
    }
    if ( s.length() > t.length() ) {
      return +1;
    }
    return 0;
  }
```

If the character at position *i* in `s` is smaller than that in `t`, return -1

StringComparison.java

```
import java.util.Scanner;
public class StringComparison {
  public static int compare( String s, String t ) {
    int leng = Math.min( s.length(), t.length() );
    for ( int i = 0; i < leng; i ++ ) {
      if ( s.charAt( i ) < t.charAt( i ) ) {
        return -1;
      }
      if ( s.charAt( i ) > t.charAt( i ) ) {
        return +1;
      }
    }
    if ( s.length() < t.length() ) {
      return -1;
    }
    if ( s.length() > t.length() ) {
      return +1;
    }
    return 0;
  }
```

If the character at position *i* in `s` is greater than that in `t`, return −1; note that `else` is not needed here

StringComparison.java

```java
import java.util.Scanner;
public class StringComparison {
  public static int compare( String s, String t ) {
    int leng = Math.min( s.length(), t.length() );
    for ( int i = 0; i < leng; i ++ ) {
      if ( s.charAt( i ) < t.charAt( i ) ) {
        return -1;
      }
      if ( s.charAt( i ) > t.charAt( i ) ) {
        return +1;
      }
    }
    if ( s.length() < t.length() ) {
      return -1;
    }
    if ( s.length() > t.length() ) {
      return +1;
    }
    return 0;
  }
```

No differences in the positions 0 .. `leng-1`; if `s` is smaller than `t`, return -1

StringComparison.java

```java
import java.util.Scanner;
public class StringComparison {
  public static int compare( String s, String t ) {
    int leng = Math.min( s.length(), t.length() );
    for ( int i = 0; i < leng; i ++ ) {
      if ( s.charAt( i ) < t.charAt( i ) ) {
        return -1;
      }
      if ( s.charAt( i ) > t.charAt( i ) ) {
        return +1;
      }
    }
    if ( s.length() < t.length() ) {
      return -1;
    }
    if ( s.length() > t.length() ) {
      return +1;
    }
    return 0;
  }
```

If `s` is greater than `t`, return $+1$; no `else` is needed

StringComparison.java

```java
import java.util.Scanner;
public class StringComparison {
  public static int compare( String s, String t ) {
    int leng = Math.min( s.length(), t.length() );
    for ( int i = 0; i < leng; i ++ ) {
      if ( s.charAt( i ) < t.charAt( i ) ) {
        return -1;
      }
      if ( s.charAt( i ) > t.charAt( i ) ) {
        return +1;
      }
    }
    if ( s.length() < t.length() ) {
      return -1;
    }
    if ( s.length() > t.length() ) {
      return +1;
    }
    return 0;
  }
```

`s` and `t` are identical, so return 0; no `else` here either

StringComparison.java (cont'd)

```
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter string 1 and hit return: " );
    String s1 = console.nextLine();
    System.out.print( "Enter string 2 and hit return: " );
    String s2 = console.nextLine();
    int result = compare( s1, s2 );
    if ( result == 0 ) {
      System.out.println( "They are equal to each other." );
    }
    else if ( result < 0 ) {
      System.out.println( "No.1 is smaller than no.2." );
    }
    else {
      System.out.println( "No.1 is greater than no.2." );
    }
  }
}
```

Define scanner

StringComparison.java (cont’d)

```
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter string 1 and hit return: " );
    String s1 = console.nextLine();
    System.out.print( "Enter string 2 and hit return: " );
    String s2 = console.nextLine();
    int result = compare( s1, s2 );
    if ( result == 0 ) {
      System.out.println( "They are equal to each other." );
    }
    else if ( result < 0 ) {
      System.out.println( "No.1 is smaller than no.2." );
    }
    else {
      System.out.println( "No.1 is greater than no.2." );
    }
  }
}
```

Receive the first string

StringComparison.java (cont'd)

```java
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter string 1 and hit return: " );
    String s1 = console.nextLine();
    System.out.print( "Enter string 2 and hit return: " );
    String s2 = console.nextLine();
    int result = compare( s1, s2 );
    if ( result == 0 ) {
      System.out.println( "They are equal to each other." );
    }
    else if ( result < 0 ) {
      System.out.println( "No.1 is smaller than no.2." );
    }
    else {
      System.out.println( "No.1 is greater than no.2." );
    }
  }
}
```

Receive the second string

StringComparison.java (cont’d)

```
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter string 1 and hit return: " );
    String s1 = console.nextLine();
    System.out.print( "Enter string 2 and hit return: " );
    String s2 = console.nextLine();
    int result = compare( s1, s2 );
    if ( result == 0 ) {
      System.out.println( "They are equal to each other." );
    }
    else if ( result < 0 ) {
      System.out.println( "No.1 is smaller than no.2." );
    }
    else {
      System.out.println( "No.1 is greater than no.2." );
    }
  }
}
```

Call the comparison method and store the result

StringComparison.java (cont’d)

```java
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter string 1 and hit return: " );
    String s1 = console.nextLine();
    System.out.print( "Enter string 2 and hit return: " );
    String s2 = console.nextLine();
    int result = compare( s1, s2 );
    if ( result == 0 ) {
      System.out.println( "They are equal to each other." );
    }
    else if ( result < 0 ) {
      System.out.println( "No.1 is smaller than no.2." );
    }
    else {
      System.out.println( "No.1 is greater than no.2." );
    }
  }
}
```

The case in which the result is 0

StringComparison.java (cont'd)

```java
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );
    System.out.print( "Enter string 1 and hit return: " );
    String s1 = console.nextLine();
    System.out.print( "Enter string 2 and hit return: " );
    String s2 = console.nextLine();
    int result = compare( s1, s2 );
    if ( result == 0 ) {
      System.out.println( "They are equal to each other." );
    }
    else if ( result < 0 ) {
      System.out.println( "No.1 is smaller than no.2." );
    }
    else {
      System.out.println( "No.1 is greater than no.2." );
    }
  }
}
```

The case in which the result is negative

StringComparison.java (cont’d)

```
 public static void main( String[] args ) {
  Scanner console = new Scanner( System.in );
  System.out.print( "Enter string 1 and hit return: " );
  String s1 = console.nextLine();
  System.out.print( "Enter string 2 and hit return: " );
  String s2 = console.nextLine();
  int result = compare( s1, s2 );
  if ( result == 0 ) {
   System.out.println( "They are equal to each other." );
  }
  else if ( result < 0 ) {
   System.out.println( "No.1 is smaller than no.2." );
  }
  else {
   System.out.println( "No.1 is greater than no.2." );
  }
 }
}
```

The case in which the result is positive

Find All Integer Sequences in an Input Line

Prompt the user and receive an input sequence using `nextLine`, then identify in the input string maximally long character sequences that can be interpreted as integers, where a number may start with a ’-’

For example, input "abc-456-78abc23" has three integer sequences: -456, -78, and 23

A Clever Solution

- Attach one whitespace at the end of input
- Use three variables for recording information:
  - `hasDigit`: boolean; records whether we have seen a digit since the last output of a number a digit; initially false
  - `hasMinus`: boolean; records whether we have seen a minus sign possibly preceding a digit; initially false
  - `aNumber`: the number that the digits that have been read represent; initially 0

Strategy

Read the letters of the input from start to end; let `c` be the current letter

1. If `c` a digit, update `aNumber` with the current value * 10 + the number the digit represent and set `hasDigit` to true
2. Otherwise, if `c` is '-' and `hasDigit` is false, then set `hasMinus` to true (to anticipate for digits to come)
3. Otherwise, produce the `aNumber` on screen; before that, multiply `aNumber` by -1 if `hasMinus` is true; after producing output, re-initialize `hasDigit` and `aNumber`; re-initialize `hasMinus` with whether `c` is '-'

FindInt.java (part 1)

```java
import java.util.Scanner;
//--------- find all integers appearing in an input String
public class FindInt {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );

    System.out.println( "Enter a line that may contain numbers" );
    System.out.print( ">> " );
    String line = console.nextLine() + " ";
    //--- variables and initialization
    int aNumber = 0;
    boolean hasMinus = false;
    boolean hasDigit = false;

    //--- the loop
    for ( int i = 0; i <= line.length() - 1; i ++ ) {
      char c = line.charAt( i );
      System.out.printf( "%03d:’%s’%n", i, "" + c );
```

Scanner creation; prompting the user

FindInt.java (part 1)

```java
import java.util.Scanner;
//-------- find all integers appearing in an input String
public class FindInt {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );

    System.out.println( "Enter a line that may contain numbers" );
    System.out.print( ">> " );
    String line = console.nextLine() + " ";
    //--- variables and initialization
    int aNumber = 0;
    boolean hasMinus = false;
    boolean hasDigit = false;

    //--- the loop
    for ( int i = 0; i <= line.length() - 1; i ++ ) {
      char c = line.charAt( i );
      System.out.printf( "%03d:'%s'%n", i, "" + c );
```

Reading the input line and attach a ' ' at the end

FindInt.java (part 1)

```java
import java.util.Scanner;
//-------- find all integers appearing in an input String
public class FindInt {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );

    System.out.println( "Enter a line that may contain numbers" );
    System.out.print( ">> " );
    String line = console.nextLine() + " ";
    //--- variables and initialization
    int aNumber = 0;
    boolean hasMinus = false;
    boolean hasDigit = false;

    //--- the loop
    for ( int i = 0; i <= line.length() - 1; i ++ ) {
      char c = line.charAt( i );
      System.out.printf( "%03d:'%s'%n", i, "" + c );
```

Variable initialization

FindInt.java (part 1)

```java
import java.util.Scanner;
//-------- find all integers appearing in an input String
public class FindInt {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );

    System.out.println( "Enter a line that may contain numbers" );
    System.out.print( ">> " );
    String line = console.nextLine() + " ";
    //--- variables and initialization
    int aNumber = 0;
    boolean hasMinus = false;
    boolean hasDigit = false;

    //--- the loop
    for ( int i = 0; i <= line.length() - 1; i ++ ) {
      char c = line.charAt( i );
      System.out.printf( "%03d:'%s'%n", i, "" + c );
```

For loop

FindInt.java (part 1)

```java
import java.util.Scanner;
//-------- find all integers appearing in an input String
public class FindInt {
  public static void main( String[] args ) {
    Scanner console = new Scanner( System.in );

    System.out.println( "Enter a line that may contain numbers" );
    System.out.print( ">> " );
    String line = console.nextLine() + " ";
    //--- variables and initialization
    int aNumber = 0;
    boolean hasMinus = false;
    boolean hasDigit = false;

    //--- the loop
    for ( int i = 0; i <= line.length() - 1; i ++ ) {
      char c = line.charAt( i );
      System.out.printf( "%03d:'%s'%n", i, "" + c );
```

Update the char variable `c` and print its value with the position value

FindInt.java (part 2)

```
      if ( Character.isDigit( c ) ) {
        aNumber = aNumber * 10 + Character.getNumericValue( c );
        hasDigit = true;
      }
      else if ( c == '-' && !hasDigit ) {
        hasMinus = true;
      }
      else {
        if ( hasDigit ) {
          if ( hasMinus ) {
            aNumber *= -1;
          }
          System.out.printf( " Found number %d%n", aNumber );
        }
        aNumber = 0;
        hasDigit = false;
        hasMinus = ( c == '-' );
      }
    }
  }
}
```

If `c` is a digit, update `aNumber`; the program assumes that the value of `aNumber` is 0 prior to the arrival of a digit

Also, set `hasDigit` to true

FindInt.java (part 2)

```java
      if ( Character.isDigit( c ) ) {
        aNumber = aNumber * 10 + Character.getNumericValue( c );
        hasDigit = true;
      }
      else if ( c == '-' && !hasDigit ) {
        hasMinus = true;
      }
      else {
        if ( hasDigit ) {
          if ( hasMinus ) {
            aNumber *= -1;
          }
          System.out.printf( " Found number %d%n", aNumber );
        }
        aNumber = 0;
        hasDigit = false;
        hasMinus = ( c == '-' );
      }
    }
  }
}
```

If `c` is a minus and the digit has not been seen yet, set `hasMinus` to true

FindInt.java (part 2)

```java
      if ( Character.isDigit( c ) ) {
        aNumber = aNumber * 10 + Character.getNumericValue( c );
        hasDigit = true;
      }
      else if ( c == '-' && !hasDigit ) {
        hasMinus = true;
      }
      else {
        if ( hasDigit ) {
          if ( hasMinus ) {
            aNumber *= -1;
          }
          System.out.printf( " Found number %d%n", aNumber );
        }
        aNumber = 0;
        hasDigit = false;
        hasMinus = ( c == '-' );
      }
    }
  }
}
```

The remaining case falls in here

FindInt.java (part 2)

```java
      if ( Character.isDigit( c ) ) {
        aNumber = aNumber * 10 + Character.getNumericValue( c );
        hasDigit = true;
      }
      else if ( c == '-' && !hasDigit ) {
        hasMinus = true;
      }
      else {
        if ( hasDigit ) {
          if ( hasMinus ) {
            aNumber *= -1;
          }
          System.out.printf( " Found number %d%n", aNumber );
        }
        aNumber = 0;
        hasDigit = false;
        hasMinus = ( c == '-' );
      }
    }
  }
}
```

If `hasDigit` is true, it is indeed a digit sequence we have just seen

FindInt.java (part 2)

```
      if ( Character.isDigit( c ) ) {
        aNumber = aNumber * 10 + Character.getNumericValue( c );
        hasDigit = true;
      }
      else if ( c == '-' && !hasDigit ) {
        hasMinus = true;
      }
      else {
        if ( hasDigit ) {
          if ( hasMinus ) {
            aNumber *= -1;
          }
          System.out.printf( " Found number %d%n", aNumber );
        }
        aNumber = 0;
        hasDigit = false;
        hasMinus = ( c == '-' );
      }
    }
  }
}
```

If `hasMinus` is true, then change the sign

FindInt.java (part 2)

```
      if ( Character.isDigit( c ) ) {
        aNumber = aNumber * 10 + Character.getNumericValue( c );
        hasDigit = true;
      }
      else if ( c == '-' && !hasDigit ) {
        hasMinus = true;
      }
      else {
        if ( hasDigit ) {
          if ( hasMinus ) {
            aNumber *= -1;
          }
          System.out.printf( " Found number %d%n", aNumber );
        }
        aNumber = 0;
        hasDigit = false;
        hasMinus = ( c == '-' );
      }
    }
  }
}
```

Output the number

FindInt.java (part 2)

```
      if ( Character.isDigit( c ) ) {
        aNumber = aNumber * 10 + Character.getNumericValue( c );
        hasDigit = true;
      }
      else if ( c == '-' && !hasDigit ) {
        hasMinus = true;
      }
      else {
        if ( hasDigit ) {
          if ( hasMinus ) {
            aNumber *= -1;
          }
          System.out.printf( " Found number %d%n", aNumber );
        }
        aNumber = 0;
        hasDigit = false;
        hasMinus = ( c == '-' );
      }
    }
  }
}
```

Reinitialize

The End