Lab07
Java Strings

Submission Details

In this lab, you are required to write **THREE** Java programs to solve the given problems shown in the section "Lab Exercises". To make the program implementation easier, you are suggested to write your program using **eclipse** instead of doing it directly using paper & pencil. After you have completed the implementation using **eclipse**, submit your program files (i.e. WelcomeString.java, CourseSplitter.java and SumASCIICode.java) to the assignment page of Canvas. The deadline would be **ONE** week after your lab section is conducted. You are reminded to double-check your solution to verify everything in correct before submission.

Note: You are required to put your name, student ID and lab section at the beginning of your source file.

Important: Only answer of selected question will be graded.

Objectives

The objective of this lab is to (1) give you a brief review on what you should have already learned in AST10106 Introduction to Programming, and (2) give you some hands-on practice on using Java packages, in particular, Java String class for this lab.

Software Required

The following software is required for this lab.

1. Java Development Kit, eg JavaSE 1.8
2. An Integrated Development Environment, e.g. eclipse

Introduction

In this lab, you will again practice:

1. Forming a program skeleton by declaring a class with a main method / operation
2. Declaring variables to store values in Java programs
3. Obtaining inputs from keyboard using methods / operations in the Scanner class
4. Solving problems by performing string operations
5. Displaying the solution of problems on screen using methods / operations in the System class

Background

1. **Form a program skeleton by defining a class with a main method / operation**

   - **Syntax:**

```
// The filename of the Java source file should be exactly the same
// as the <class name>
public class <class name>
{
    // Define a main method / operation, which is the starting point of the program
    public static void main(String[] args)
    {
       // Program statements here
    }
}
```

2. **Declare variables for storing values**

   - **Syntax:**

```
// Declare a variable with name <name of variable> in type <type>
<type> <name of variable>;
// Declare a variable with name <name of variable> in type <type> with
// initial value <initial value>
<type> <name of variable> = <initial value>;
// Declare a number of variables with name <variable name1>,
// <variable name2>, … all in type <type>
<type> <variable name1>, <variable name2>, …;
// Declare a number of variables with name <variable name1>,
// <variable name2>, … all in type <type> with initial value <initial value1>,
// <initial value2>, …, respectively
<type> <variable name1> = <initial value1>, <variable name2> = <initial value2>, …;
where <type> could be in one of the following types:
byte, short, int, long, float, double, char, boolean
```

3. **Obtain inputs from keyboard and store them in variables:**

   Syntax:

```
// Get the class definition of the Scanner class from java.util package
import java.util.Scanner;
// Create a Scanner object with name <object name>
// System.in is the “standard” input stream, which is keyboard
Scanner <object name> = new Scanner(System.in);
// Use the Scanner class’s operations (methods) to read input
<variable name> = <object name>.<method name>();
```

4. **Perform string manipulation:**

- **Syntax:**

```
// Returns the length of this string.
// The length is equal to the number of characters in the string.
public int length()

// Returns the char value at the specified index
// An index ranges from 0 to length() – 1.
// Note: The first character with index = 0, the next at index = 1, and so on.
public char charAt( int index )

// Compares this string to the specified string.
// The result is true if and only if the argument represents the same sequence
// of characters as this object.
public boolean equals( String anotherString )

// Computes two strings lexicographically
// (1) Result is negative integer if this String object lexicographically precedes
//      the argument string
// (2) Result is positive integer if this String object lexicographically follows
//      the argument string
// (3) Result is zero if the strings are equal
public int compareTo( String anotherString )

// Computes two strings lexicographically, ignoring case differences
public int compareToIgnoreCase( String anotherString )

// Return a new String object representing the concatenation of the character sequence
// represented by this String object and the character sequence represented by the argument
// string
public String concat( String anotherString )

// Return the index within this string of the FIRST occurrence of the specified character
public int indexOf( int ch )

// Return the index within this string of the FIRST occurrence of the specified character,
// starting the search at the specified index
public int indexOf( int ch, int fromIndex )

// Return the index within this string of the LAST occurrence of the specified character
public int lastIndexOf( int ch )

// Return the index within this string of the LAST occurrence of the specified character,
// starting the search at the specified index
public int lastIndexOf( int ch, int fromIndex )
```

```
// Return the index within this string of the FIRST occurrence of the specified substring
public int indexOf( String str )

// Return the index within this string of the FIRST occurrence of the specified substring,
// starting the search at the specified index
public int indexOf( String str, int fromIndex )

// Return the index within this string of the LAST occurrence of the specified substring
public int lastIndexOf( String str )

// Return the index within this string of the LAST occurrence of the specified substring,
// starting the search at the specified index
public int lastIndexOf( String str, int fromIndex )

// Return a new string that is a substring of this string.
// The substring begins with the character at the specified index and
// extends to the end of this string
public String substring( int beginIndex )

// Return a new string that is a substring of this string.
// The substring begins with the character at the specified index and extends
// to the character at index endIndex – 1. Thus the length of the substring is
// endIndex - beginIndex
public String substring( int beginIndex, int endIndex )

// Convert all of the characters in this String to lower case
public String toLowerCase()

// Convert all of the characters in this String to upper case
public String toUpperCase()

// Return the string representation of the boolean argument
public static String valueOf( boolean b )

// Return the string representation of the char argument
public static String valueOf( char c )

// Return the string representation of the int argument
public static String valueOf( int I )

// Return the string representation of the long argument
public static String valueOf( long l )

// Return the string representation of the float argument
public static String valueOf( float f )

// Return the string representation of the double argument
public static String valueOf( double d )
```

5. Display information on screen:

- **Syntax:**

```
// Print data on screen and the cursor will stay at the end of the string printed
System.out.print(<data> [+ <data>]);
// Print the data on screen and move the cursor to the beginning of the next line
System.out.println(<data> [+ <data>]);
// Print nothing
System.out.print("");
// Print nothing, but move the cursor to the beginning of the next line
System.out.println("");
```

Example:

Write a Java program to generate a password for a student using his initials and age.

Name your program as PasswordMaker.java.

The expected outputs of your program are as follows:

```
Enter last name: Chan
Enter first word of your first name: Tai
Enter second word of your first name: Man
Enter age: 18
Your password = tmc18
```

PasswordMaker.java

```
// Generates a password for a student using his
// initials and age

import java.util.Scanner;

public class PasswordMaker
{
   public static void main(String[] args)
   {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter last name: ");
      String lastName = sc.next();
      System.out.print("Enter first word of your first name: ");
      String firstName1 = sc.next();
      System.out.print("Enter second word of your first name: ");
      String firstName2 = sc.next();
      System.out.print("Enter age: ");
      int age = sc.nextInt();

      // Extract initials
      String initials =    firstName1.substring(0,1) +
                           firstName2.substring(0,1) +
                           lastName.substring(0,1);

      // Append age after changing the initials to lower case
      String password = initials.toLowerCase() + age;

      System.out.println("Your password = " + password);
   }
}
```

Lab Exercises

In this lab, you will be asked to solve **THREE** problems using Java.

Question 1:

Write a Java program "WelcomeString.java" that performs the following operations to this string "Welcome! This is AST10106 Introduction to Programming":

1. Convert all alphabets to capital letters and print out the result;
2. Convert all alphabets to lower-case letters and print out the result;
3. Print out the length of the string; and
4. Print out the index of the word "Programming".

The expected outputs of your program are as follows:

```
Upper case string: WELCOME! THIS IS AST10106 INTRODUCTION TO PROGRAMMING
Lower case string: welcome! this is ast10106 introduction to programming
Length of string: 53
Index of word Programming: 42
```

Question 2:

A CCCU course can be described using three components: <Division title>□<Code>□<Course description>, where □ represents an empty space. Write a Java program "CourseSplitter.java" that obtains a course string, and then splits and prints the **THREE** components of the course.

The expected outputs of your program are as follows:

```
Enter course string: AST 10106 Introduction to Programming
Division: AST
Course Code: 10106
Course Description: Introduction to Programming

Enter course string: BUS 10112 Introduction to Business Logistics
Division: BUS
Course Code: 10112
Course Description: Introduction to Business Logistics

Enter course string: LAC 22481 Translation: English and Chinese
Division: LAC
Course Code: 22481
Course Description: Translation: English and Chinese
```

Question 3:

Write a Java program "SumASCIICode.java" that sums the ASCII codes for all the letters of any 4-character string (e.g. java, have, code, this, …). Use the charAt() method to accomplish your work.

The expected outputs of your program are as follows:

```
Enter a 4-character string: java
Sum: 418

Enter a 4-character string: code
Sum: 411

Enter a 4-character string: this
Sum: 440
```

Marking Scheme

The marking of this exercise will be based on the following criteria.

Graded items	Weighting
1. Correctness of program (i.e. whether your code is implemented in a way according to the requirements as specified.)	50%
2. Presentation of the Java codes (i.e. whether the program is properly indented, how close you follow the common conventions as mentioned in class, etc.)	30%
3. Documentation (with reasonable amount of comments embedded in the code to enhance the readability)	20%
	100%

Be aware of plagiarism! DON'T copy the program file from your friends or classmates. If any identical copy is found, 5% of the coursework marks will be deducted for each of the file owner.

Program Submission Checklist

Before submitting your work, please check the following items to see you have done a decent job.

Items to be checked	☑ / ☒
1. Did I put my name, student ID and lab section at the beginning of all the source files as comment?	☐
2. Did I put reasonable amount of comments to describe my program?	☐
3. Are they all in .java extension and named according to the specification?	☐
4. Have I checked that all the submitted code are compliable and run without any errors?	☐
5. Did I zip my source files using Winzip / zip provided by Microsoft Windows? Also, did I check the zip file and see if it could be opened? ***(Only applicable if the work has to be submitted in zip format.)***	☐
6. Did I submit my lab assignment to Canvas before the deadline?	☐