Unit6 : HW4 on Interfaces

Any class that implements the Java’s ***CharSequence*** interface must override the following abstract methods: ***charAt***, ***length*** and ***subSequence***. For example, the class ***String*** implements ***CharSequence***. That is why you see the three methods (***charAt***, ***length*** and ***subSequence***) appear (as overriden methods) in the list of public methods of the class ***String***.

In this HW, you will be asked to implement the built-in Java interface ***CharSequence***. You will be provided with two classes: The ***DriverClass*** class and the ***Code*** class that implements ***CharSequence*** (from the util package). The class ***Code*** has ***int [][] codeArray***, ***int numRows***, ***int numColumns*** as its private fields, and it is already equipped with the code that loads and print the array ***codeArray***. Your task is to override the methods ***charAt***, ***length*** and ***subSequence*** as indicated below***:***

Assume that the array ***codeArray*** is:

125 53 63 63

78 59 71 45

69 90 95 93

Then override the :

1. ***char charAt (int index)*** to return the character whose ascii/Unicode code is at index. Given the array above, if index is 6, the element of the array code to consider is **71**, and the character to return is ***G*** (the ascci code for the **G** is **71**).

Want to check? System.***out***.println((**char**) 71);

1. ***int length()*** to return how many characters are loaded in the array (which is the number of integers in the array). In the case of the array provided here, length returns ***12***
2. ***String subSequence(int start, int end)*** returns the string formed by concatenating all the characters corresponding to the integers from index ***start*** to index ***end***. For example **subsequence(1,5)** returns ***5??N;***

A scatelon code is provided at the end of this HW. Once again, your task is to provide the code for the three methods: ***charAt***, ***length*** and ***subSequence***. (Do not change the Driver class!)

**Sample Run 1:**

**Assume that the user will enter numbers between 1 and 20**

Enter how many rows and how many columns to load: 3 4

Enter row 1: **125 53 63 63**

Enter row 2: **78 59 71 45**

Enter row 3: **69 90 95 93**

Testing charAt: Enter your index [a number greater or equal to 0 and less or equal to 11]: **6**

The character located at index 6 is: G

**Assume that the user will enter a numbers between 0 and (number of rows times number of columns – 1)**

Testing length: there are 12 characters.

Testing subSequence: Enter start and end indexes: **1 5**

The subsequuence is: 5??N;

**Assume that the user will enter two non-negative integers sunch that the first one is less or equal to the second one, and that both of the entered numbers are between 0 and (number of rows times number of columns – 1)**

Goodbye!

**Sample Run 2:**

Enter how many rows and how many columns to load: 2 8

Enter row 1: **195 63 63 63 69 92 85 95**

Enter row 2: **78 59 71 45** **85 67 70 33**

Testing charAt: Enter your index [a number greater or equal to 0 and less or equal to 11]: **0**

The character located at index 0 is: Ã

Testing length: there are 16 characters.

Testing subSequence: Enter start and end indexes: **12** **15**

The subsequuence is: UCF!

Goodbye!

**import** java.util.Scanner;

**public** **class** DriverClass {

**public** **static** **void** main(String args[]) {

**int** numRows, numColumns;

**int** index, start, end;

**char** charAtIndex;

**int** length;

String subSequence;

Scanner myScan = **new** Scanner(System.***in***);

System.***out***.print("Enter how many rows and how many columns to load: ");

numRows = myScan.nextInt();

numColumns = myScan.nextInt();

Code codeObject = **new** Code(numRows, numColumns);

codeObject.loadCodeArray(numRows, numColumns);

codeObject.printCodeArray(numRows, numColumns);

// \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

System.***out***.print("\n\nTesting charAt: Enter your index [a number greater or equal to 0 and less or equal to ");

System.***out***.print((numRows \* numColumns - 1) + "]:");

index = myScan.nextInt();

charAtIndex = codeObject.charAt(index);

System.***out***.println("The character located at index " + index + " is: " + charAtIndex);

// \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

length = numRows \* numColumns;

System.***out***.println("\n\nTesting length: there are " + length + " characters.");

// \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

System.***out***.print("\n\nTesting subSequence: Enter start and end indexes: ");

start = myScan.nextInt();

end = myScan.nextInt();

subSequence = codeObject.subSequence(start, end);

System.***out***.println("The subsequuence is: " + subSequence);

// \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

System.***out***.println("\nGoodbye!");

}

}

//\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**class** Code **implements** CharSequence {

**private** **int**[][] codeArray;

**private** **int** numRows, numColumns;

**public** Code(**int** numRows, **int** numColumns) {

**this**.numRows = numRows;

**this**.numColumns = numColumns;

codeArray = **new** **int**[numRows][numColumns];

}

**public** **void** loadCodeArray(**int** numRows, **int** numColumns) {

Scanner myScan = **new** Scanner(System.***in***);

**int** i, j;

**for** (i = 0; i < numRows; i++) {

System.***out***.print("Enter Row " + (i + 1) + ": ");

**for** (j = 0; j < numColumns; j++) {

codeArray[i][j] = myScan.nextInt();

}

}

}

**public** **void** printCodeArray(**int** numRows, **int** numColumns) {

**int** i, j;

System.***out***.println("\n\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

**for** (i = 0; i < numRows; i++) {

**for** (j = 0; j < numColumns; j++) {

System.***out***.print(codeArray[i][j] + "\t");

}

System.***out***.println("");

}

}

// \_\_\_\_\_\_\_\_\_\_\_\_\_\_THE CODE ABOVE IS TO REMAIN UNCHANGED\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

@Override

**public** **char** charAt(**int** index) {

// Add your code here...

}

@Override

**public** **int** length() {

// Add your code here...

}

@Override

**public** String subSequence(**int** start, **int** end) {

// Add your code here...

}

}