Practice Problem

Q1.Given a string and an index, return a string length 2 starting at the given index. If the index is too big or too small to define a string length 2, use the first 2 chars. The string length will be at least 2.

twoChar("java", 0) → "ja"

twoChar("java", 2) → "va"

twoChar("java", 3) → "ja"

|  |
| --- |
| **public** **static** String exe1(String val, **int** index) {      **if**( index>=val.length()-1)  **return** val.substring(0,2);  **else**  **return** val.substring(index,index+2);  } |

Q2.Given a string of odd length, return the string length 3 from its middle, so "Candy" yields "and". The string length will be at least 3.

middleThree("Candy") → "and"

middleThree("and") → "and"

middleThree("solving") → "lvi"

|  |
| --- |
| **public** **static** String midVal(String oddLenStr) {  **int** len =oddLenStr.length()/2;  **if**(oddLenStr.length()>3)  **return** oddLenStr.substring(len-1, len+2);  **else**  **return** oddLenStr.substring(0, 3);  } |

Q3.Given two strings, append them together (known as "concatenation") and return the result. However, if the concatenation creates a double-char, then omit one of the chars, so "abc" and "cat" yields "abcat".

conCat("abc", "cat") → "abcat"

conCat("dog", "cat") → "dogcat"

conCat("abc", "") → "abc"

|  |
| --- |
| **public** **static** String concCatVal(String val1, String val2) {  **if**(val2=="")  **return** val1;  **else** {  **if**(val1.charAt(val1.length()-1)==val2.charAt(0))  val1=val1.substring(0, val1.length()-1);  **return** val1.concat(val2);  }} |

Q4.Given an array of ints of odd length, return a new array length 3 containing the elements from the middle of the array. The array length will be at least 3.

midThree([1, 2, 3, 4, 5]) → [2, 3, 4]

midThree([8, 6, 7, 5, 3, 0, 9]) → [7, 5, 3]

midThree([1, 2, 3]) → [1, 2, 3]

|  |
| --- |
| **public** **static** **int**[] midElem(**int**[] oddLenArr) {  **int**[] newArr = **new** **int**[3];  **int** len =oddLenArr.length/2;  newArr[0]= oddLenArr[len-1];  newArr[1]=oddLenArr[len];;  newArr[2]=oddLenArr[len+1];;  **return** newArr;  } |

Q5.Given an int array, return true if the array contains 2 twice, or 3 twice. The array will be length 0, 1, or 2.

double23([2, 2]) → true

double23([3, 3]) → true

double23([2, 3]) → false

|  |
| --- |
| **public** **static** **boolean** double23(**int**[]smeElem2or3) {    **if**(smeElem2or3.length>1 &&  (smeElem2or3[0]==2&&smeElem2or3[1]==2  ||(smeElem2or3[0]==3 &&smeElem2or3[1]==3)))  **return** **true**;  **else**  **return** **false**;  } |

Q6.Write a program to print the pattern given below (Left Triangle Star Pattern)

\*

\* \*

\* \* \*

\* \* \* \*

|  |
| --- |
| **public** **static** **void** printPattern(**int** n) {  **int** i,j;  **for**( i=0;i<n;i++) {  **for**( j =0; j<=i; j++) {  System.***out***.print("\*"+" ");  }  System.***out***.println();  }    } |

\* \* \* \* \*

Q7.Write a program to print Fibonacci Series up to count 10.

Fibonacci Series example: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55…

|  |
| --- |
| **public** **static** **void** FibonacciSeries(**int** count) {    **int** num1 = 0;  **int** num2 = 1;  **int** sum = 0;  System.***out***.print("Fibonacci Series upto count "+ count+ " is : ");  **for**(**int** i=1; i<=count;i++) {    System.***out***.print(sum+" ");  num1=num2;  num2 = sum;  sum = num1 + num2;  }  } |

Q8 Find The Largest Value From The Given Array.

|  |
| --- |
| **public** **static** **int** LarArrVal(**int**[] arr) {    **int** max=arr[0];  **for**( **int** i =0; i<arr.length;i++) {  **for**(**int** j= i+1;j<arr.length;j++) {  **if**(arr[i]>arr[j]) {  max=arr[i];  arr[i]=arr[j];  arr[j]=max;  }  }    }  Arrays.*sort*(arr);  System.***out***.println("Largest value using Arrays.sort is "+( arr[arr.length-1]));  **return** max;  } |

Q9.How to display all the prime numbers between 1 and 100.

|  |
| --- |
| **public** **static** **void** PrimeNum(**int** NumUpto) {    **int** count=0;  **int** i;  System.***out***.println("The Prime numbers upto "+ NumUpto+ " are: ");  **for**( i=1; i<=NumUpto;i++) {  count =0;  **for**(**int** j=1;j<=i;j++) {  **if**(i%j==0)  count++;  }  **if**(count == 2) {  System.***out***.print(i+" ");  }  }    } |

|  |
| --- |
| **int** count=0;  **char**[] valArr =val.toCharArray();  **for**(**char** e :valArr) {  **if**(present==e)  count++;  }    **return** count;  } |

Q10.Take in a String from user and take in a character from user. Once you get them, print out how many times the given character is present in the given String.