|  |  |  |
| --- | --- | --- |
| 1. Refer to the code below to answer the following   String s = “Get here Thanksgiving!”;  String m = “er”;  int j = 8, z = 99; | | |
| (a)  int k = s.indexOf(m);  System.out.println(k); | **5** | |
| (b)  int k = s.indexOf(‘T’);  System.out.println(k); | **9** | |
| (c)  char p = s.charAt(6);  System.out.println(p); | **r** | |
| (d)  int k = s.indexOf(z);  System.out.println(k); | **-1** | |
| (e)  int k = s.indexOf(‘g’, j);  System.out.println(k); | **15** | |
| (f)  char p = s.charAt(z – 90);  System.out.println(p); | **T** | |
| (g)  int k = s.indexOf(m, 15);  System.out.println(k); | **-1** | |
| (h)  int k = s.indexOf(z + 2, 4);  System.out.println(k); | **5** | |
| (i)  boolean k = s.contains(m);  System.out.println(k); | **true** | |
| (j)  String s2 = “ JAVA “;  String k = “!” + s2.trim() + “!”  System.out.println(k); | **!JAVA!** | |
| (k)  System.out.println(m.compareTo(s)); | **30** | |
|  | | /11 |

|  |
| --- |
| 1. The Alphabetize class below, alphabetizes three words. Consider the following examples. Write the Alphabetize class. |
| |  |  | | --- | --- | | **Values of Strings s1, s2, and s3 before** | **Values of s1, s2, and s3 after** | | String s1 = “cat”;  String s2 = “car”;  String s3 = “dog”; | String s1 = “car”;  String s2 = “cat”;  String s3 = “dog”; | | String s1 = “dog”;  String s2 = “cat”;  String s3 = “car”; | String s1 = “car”;  String s2 = “cat”;  String s3 = “dog”; | |
| public class Alphabetize{  public static void main(String args[]){    **//check if s1 is last**  **if(s1.compareTo(s2)>0 && s1.compareTo(s3)>0){**  **temp = s3;**  **s3 = s1;**  **s1 = temp;**  **}**  **//check if s2 is last**  **if(s2.compareTo(s1)>0 && s2.compareTo(s3)>0){**  **temp = s3;**  **s3 = s2;**  **s2 = temp;**  **}**  **//compare s1 and s2**  **if(s1.compareTo(s2)>0){**  **temp = s2;**  **s2 = s1;**  **s1 = temp;**  **}**  **System.out.println(s1 + " " + s2 + " " + s3);**  }  } |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. Write an algorithm that could be used to count the number of times a string occurs in another string. Consider the examples below[[1]](#footnote-1). This algorithm requires that you incorporate a loop along with the substring() and length() methods.  |  |  |  | | --- | --- | --- | | **String to search** | **String to find** | **Occurrences** | | BAAB | AA | 1 | | AAAAA | AA | 2 | | AABABABAA | ABA | 2 | | ABBAABB | ABA | 0 | |
| public class FindOccur{  public static void main(String args[]){  **Solution 1**  code Line 1: public static int countRepeat(String largeStr, String smallStr) Line 2: { Line 3: int count = 0; Line 4: int i = 0; Line 5: int len = smallStr.length(); Line 6 is blank. Line 7: while (i < largeStr.length() – len + 1) Line 8: { Line 9: if (smallStr.equals(largeStr.substring(i, i + len))) Line 10: { Line 11: count++; Line 12: i += len; Line 13: } Line 14: else Line 15: { Line 16: i++; Line 17: } Line 18: } Line 19: return count; Line 20: } end code    **Solution 2**    }  } |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. The *removeSub* class below removes string from another string and composes the result. If the substring is not found the original string is returned. Consider the examples below,  |  |  |  | | --- | --- | --- | | **String to search** | **Substring to remove** | **Result** | | ABAAAAABAAAAABA | ABA | AAAAAA | | ABAAAAABAAAAABA | ABAA | AAAAABA | | ABAAAAABAAAAABA | AABAA | ABAAAAAABA | | ABAAAAABAAAAABA | ACA | ABAAAAABAAAAABA |   You must use the scanner methods *setDelimiter()* and *hasNext()* in your solution. | |
| public class removeSub{  public static void main(String args[]){  Scanner sc = new Scanner("ABAAAAABAAAAABA");  String delimiter = /\*some substring to be removed\*/    **String result = "";**  **sc.useDelimiter(delimiter);**  **while(sc.hasNext()){**  **result += sc.next();**  **}**    }  } | |
|  | /3 |

1. Adapted from the 2020 AP Computer Science A Exam [↑](#footnote-ref-1)