|  |  |  |
| --- | --- | --- |
| 1. What is the output of the following code? | | |
| int x = 2, p = 5, y = -8;  switch(x){  case 2:  p++;  case 3:  case 4:  y+=(p--);  break;  case 5:  y+=(p++);  }  System.out.println(y) | String s = “Red”;  int q = 0;  switch(s){  case “Red”:  q++;  case “Green”:  q++;  case “Blue”:  q++;  case “Yellow”:  q++;  default:  q++;  }  System.out.println(--q) | |
| -2 | 4 | |
|  | | /2 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. Find smallest, locates the digit with the smallest value in a number.   Below are examples,   |  |  | | --- | --- | | **Number** | **Smallest** | | 955538 | 3 | | 989811119898 | 1 | | 999999 | 9 |   Write your algorithm below. Your solution cannot use string operations and must use a for loop. | |
| public class findSmallest{  public static void main(String args[]){  int num = /\* some number \*/            smallest = num%10;          for(int i = num; i > 0; i /= 10){              int temp = i %10;              if(temp < smallest){                  smallest = temp;              }          }          System.out.println(smallest);  }  } | |
|  | /4 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. The WordManip class below removes a letter from a word and composes new word based on the remaining letters. You may assume the word and letter are all capital letter.   Below are examples,   |  |  |  | | --- | --- | --- | | **Word** | **letter** | **Result** | | ELEPHANT | E | LPHANT | | ABRACADABRA | A | BRCDBR | | HORSE | D | HORSE |   Write your algorithm below. Your solution must use a for loop. | |
| public class WordManip{  public static void main(String args[]){  String word = /\* some word \*/  String letter = /\* some letter \*/  String result = "";          for(int i = 0; i < word.length(); i++){              if(word.substring(i, i + 1).equals(letter)){                  result += "";              }else{                  result += word.substring(i, i + 1);              }          }          System.out.println(result);  }  } | |
|  | /3 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. The prime number counter below counts the number of two-digit prime numbers in a list of numbers.   Below are examples,   |  |  |  | | --- | --- | --- | | **Number** | **Number of primes** | **Prime numbers** | | 131744 | 2 | 17 13 | | 201822 | 0 |  | | 99434137 | 3 | 37 41 43 |   Complete the prime number counter below. You may assume that each number to be checked contains an even number of digits. The prime number counter must count the number of two-digit prime numbers in a number with an even number of digits and store the result in count. You must use a for loop. String operations are not allowed. | |
| public class PrimeCounter {  public static void main(String args[]){  int num = /\* a number with an even number of digits \*/  int count = 0;//count of the number of prime numbers            for(int i = num; i > 0; i = i/100){              int temp = i % 100;              boolean isPrime = true;              for(int j = 2; j < 10; j++){                  if(temp % j == 0){                      isPrime = false;                  }              }              if(isPrime){                  count++;              }          }          System.out.println(count);  }  } | |
|  | /5 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. The wordle game below evaluates whether a user guesses a secret word based on the following criteria.  * If the letter in the guess is in the same position as the letter in the secret word, the letter is displayed in the correct position * If the letter in the guess is in the secret word but in a different position, an asterisk is displayed in that position * If the letter in the guess in not in the secret word, an underscore is dispaled   Below are examples,   |  |  |  | | --- | --- | --- | | **Secret** | **Guess** | **Result** | | EARTH | HEART | \*\*\*\*\* | | EARTH | BARKS | \_AR\_ \_ | | EARTH | AMPLE | \*\_ \_ \_\* |   Complete the World class below to meet the specifications above. The result of guess must be stored in the variable result. | |
| public class Wordle {  public static void main(String args[]){  String secret = /\* some five letter word \*/;  String result = “”;  Scanner s = new Scanner(System.in);  System.out.println(“Guess my word ”);  String guess = s.next();          for (int i = 0; i < guess.length(); i++) {              String temp = guess.substring(i, i + 1);              boolean found = false;              for (int j = 0; j < secret.length(); j++) {                  if(temp.equals(secret.substring(j, j + 1))){                      found = true;                  }              }              //if the temp letter is in the same position in the guess              //we got the letter              if(temp.equals(secret.substring(i, i + 1))){                  result += temp;              //if found is true, the letter is in the word,  //but wrong location              }else if(found){                  result += "\*";              //otherwise the letter is not in the word              }else{                  result += "\_";              }              //reset false and keep checking              found = false;          }          System.out.println(result);  }  } | |
|  | /5 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. The CountSub class below counts the number of times a substring occurs in another string.   Below are examples. Note that the found substring in each string is bolded.   |  |  |  | | --- | --- | --- | | **str** | **substr** | **count** | | **ABA**BAB | ABA | 1 | | BB**ABA**B**ABA** | ABA | 2 | | **ABABABABAB**A | AB | 5 | | BBAABBAABBAA | ABA | 0 |   Complete the CountSub class below to meet the specifications above. Be mindful of boundaries and how you increment through the for loop depending on whether the substring is found or not. | |
| public class CountSub {  public static void main(String args[]){  String str = /\* some string \*/;  String substr = /\* some substring \*/  int count = 0;          for(int i = 0; i <= str.length() - substr.length(); i += 0){                if(str.substring(i, i + substr.length()).equals(substr)){                  count++;                  i += substr.length();              }else{                  i++;              }          }          System.out.println(count);  }  } | |
|  | /4 |