|  |
| --- |
| import java.util.\*; |
|  |  |
|  |  |
|  | public class Task1 { |
|  | public static void main(String[] args) { |
|  | Scanner input = new Scanner(System.in); |
|  | System.out.println("1. Encrypt\n2. Decrypt"); |
|  | String inputFunction = input.nextLine(); |
|  | if (inputFunction.equals("1")) { |
|  | System.out.printf("String to encrypt: "); |
|  | String string = input.nextLine(); |
|  | int[] result = encrypt(string); |
|  | for (int i: result) { |
|  | System.out.println(i); |
|  | } |
|  | } |
|  | if (inputFunction.equals("2")) { |
|  | System.out.printf("String to decrypt: "); |
|  | String string = input.nextLine(); |
|  | String[] stringArray = string.split(" "); |
|  | String result = decrypt(stringArray); |
|  | System.out.println(result); |
|  | } |
|  | } |
|  |  |
|  |  |
|  | public static int[] encrypt(String string) { |
|  | int[] result = new int[string.length()]; |
|  | int last = 0; |
|  | int i = 0; |
|  | for (char c: string.toCharArray()) { |
|  | result[i] = (int)c - last; |
|  | last = (int)c; |
|  | i++; |
|  | } |
|  | return result; |
|  | } |
|  |  |
|  |  |
|  | public static String decrypt(String[] string) { |
|  | String result = ""; |
|  | int last = 0; |
|  | for (String i: string) { |
|  | result += String.valueOf((char)(Integer.parseInt(i) + last)); |
|  | last += Integer.parseInt(i); |
|  | } |
|  | return result; |
|  | } |
|  | } |

|  |
| --- |
| import java.util.\*; |
|  |  |
|  |  |
|  | public class Task3 { |
|  | public static void main(String[] args) { |
|  | Scanner input = new Scanner(System.in); |
|  | System.out.printf("Shortcut: "); |
|  | String shortword = input.nextLine(); |
|  | System.out.printf("Word: "); |
|  | String longword = input.nextLine(); |
|  | System.out.println(canComplete(shortword, longword)); |
|  | } |
|  |  |
|  |  |
|  | public static boolean canComplete(String shortword, String longword) { |
|  | boolean exists = false; |
|  | for (char c: shortword.toCharArray()) { |
|  | exists = false; |
|  | for (int i = 0; i < longword.length(); i++) { |
|  | if (c == longword.charAt(i)) { |
|  | longword = longword.substring(i + 1); |
|  | exists = true; |
|  | break; |
|  | } |
|  | } |
|  | if (exists == false) return false; |
|  | } |
|  | return true; |
|  | } |
|  | } |

|  |
| --- |
| import java.util.\*; |
|  |  |
|  |  |
|  | public class Task4 { |
|  | public static void main(String[] args) { |
|  | //Scanner input = new Scanner(System.in); |
|  | //String longword = input.nextLine(); |
|  | System.out.println(sumDigProd(args)); |
|  | } |
|  |  |
|  |  |
|  | public static int sumDigProd(String... args) { |
|  | int[] nums = new int[args.length]; |
|  | for (int i = 0; i < args.length; i++) { |
|  | nums[i] = Integer.parseInt(args[i]); |
|  | } |
|  | int count = 0; |
|  | int multi = 1; |
|  | for (int num: nums) { |
|  | count += num; |
|  | } |
|  | for (char c: String.valueOf(count).toCharArray()) { |
|  | multi \*= c - '0'; |
|  | } |
|  | if (String.valueOf(multi).length() == 1) return multi; |
|  | return sumDigProd(String.valueOf(multi)); |
|  | } |
|  | } |

|  |
| --- |
| import java.util.\*; |
|  |  |
|  |  |
|  | public class Task4 { |
|  | public static void main(String[] args) { |
|  | //Scanner input = new Scanner(System.in); |
|  | //String longword = input.nextLine(); |
|  | System.out.println(sumDigProd(args)); |
|  | } |
|  |  |
|  |  |
|  | public static int sumDigProd(String... args) { |
|  | int[] nums = new int[args.length]; |
|  | for (int i = 0; i < args.length; i++) { |
|  | nums[i] = Integer.parseInt(args[i]); |
|  | } |
|  | int count = 0; |
|  | int multi = 1; |
|  | for (int num: nums) { |
|  | count += num; |
|  | } |
|  | for (char c: String.valueOf(count).toCharArray()) { |
|  | multi \*= c - '0'; |
|  | } |
|  | if (String.valueOf(multi).length() == 1) return multi; |
|  | return sumDigProd(String.valueOf(multi)); |
|  | } |
|  | } |

|  |
| --- |
| import java.util.\*; |
|  |  |
|  |  |
|  | public class Task5 { |
|  | public static void main(String[] args) { |
|  | System.out.println(sameVowelGroup(args)); |
|  | } |
|  |  |
|  |  |
|  | public static List<String> sameVowelGroup(String[] args) { |
|  | String[] allowed = new String[] {"A", "E", "Y", "U", "I", "O"}; |
|  | List<String> list = Arrays.asList(allowed); |
|  | List<String> result = new ArrayList<String>(); |
|  | String first = args.length > 0 ? args[0] : ""; |
|  | boolean good = false; |
|  | for (String str: args) { |
|  | for (char f: first.toCharArray()) { |
|  | if (list.contains(String.valueOf(f).toUpperCase())) { |
|  | for (char s: str.toCharArray()) { |
|  | if (s == f) { |
|  | good = true; |
|  | break; |
|  | } |
|  | good = false; |
|  | } |
|  | } |
|  | } |
|  | if (good) result.add(str); |
|  | } |
|  | return result; |
|  | } |
|  | } |

|  |
| --- |
| import java.util.\*; |
|  |  |
|  |  |
|  | public class Task6 { |
|  | public static void main(String[] args) { |
|  | System.out.println(validateCard(args[0])); |
|  | } |
|  |  |
|  |  |
|  | public static boolean validateCard(String card) { |
|  | if (card.length() < 16 || card.length() > 19) return false; |
|  | int checkNumber = Integer.parseInt(String.valueOf(card.charAt(card.length() - 1))); |
|  | int longNumberSum = 0; |
|  | int currentNumber = 0; |
|  | int sum = 0; |
|  | String sumString = ""; |
|  | for (int i = card.length() - 2; i >= 0; i--) { |
|  | if (i % 2 == 0) currentNumber = Integer.parseInt(String.valueOf(card.charAt(i))) \* 2; |
|  | else currentNumber = Integer.parseInt(String.valueOf(card.charAt(i))); |
|  | if (currentNumber > 9) { |
|  | longNumberSum = 0; |
|  | for (char c: String.valueOf(currentNumber).toCharArray()) { |
|  | longNumberSum += Integer.parseInt(String.valueOf(c)); |
|  | } |
|  | sum += longNumberSum; |
|  | continue; |
|  | } |
|  | sum += currentNumber; |
|  | } |
|  | sumString = String.valueOf(sum); |
|  | return (10 - Integer.parseInt(String.valueOf((sumString.charAt(sumString.length() - 1)))) == checkNumber); |
|  | } |
|  | } |

|  |
| --- |
| import java.util.\*; |
|  |  |
|  |  |
|  | public class Task7 { |
|  | public static void main(String[] args) { |
|  | System.out.println(numToEng(Integer.parseInt(args[0]))); |
|  | } |
|  |  |
|  |  |
|  | public static String numToEng(int num) { |
|  | if (num == 0) return "zero"; |
|  | String result = ""; |
|  | String[] firstNumbers = new String[] {"", "one", "two", "three", "four", "five", "six", "seven", |
|  | "eight", "nine", "ten", "eleven", "twelve", "thirteen", "fourteen", |
|  | "fifteen", "sixteen", "seventeen", "eighteen", "nineteen"}; |
|  | String[] secondNumbers = new String[] { |
|  | "", // 0 |
|  | "", // 1 |
|  | "twenty", // 2 |
|  | "thirty", // 3 |
|  | "forty", // 4 |
|  | "fifty", // 5 |
|  | "sixty", // 6 |
|  | "seventy", // 7 |
|  | "eighty", // 8 |
|  | "ninety" // 9 |
|  | }; |
|  | if (num < 20) { |
|  | result = firstNumbers[num]; |
|  | } |
|  | else if (num < 100) { |
|  | result = secondNumbers[num / 10] + (num % 10 != 0 ? " ": "") + firstNumbers[num % 10]; |
|  | } |
|  | else if (num < 1000) { |
|  | result = firstNumbers[num / 100] + " hundered " + (num % 100 != 0 ? numToEng(num % 100) : ""); |
|  | } |
|  | return result; |
|  | } |
|  | } |

|  |
| --- |
| import java.util.\*; |
|  | import java.security.MessageDigest; |
|  | import java.nio.charset.StandardCharsets; |
|  | import java.security.NoSuchAlgorithmException; |
|  |  |
|  |  |
|  |  |
|  | public class Task8 { |
|  | public static void main(String[] args) throws NoSuchAlgorithmException { |
|  | Scanner input = new Scanner(System.in); |
|  | System.out.printf("What do you want to hash?: "); |
|  | String unhased = input.nextLine(); |
|  | System.out.println(getHash(unhased)); |
|  | } |
|  |  |
|  |  |
|  | public static String getHash(String str) throws NoSuchAlgorithmException { |
|  | MessageDigest md = MessageDigest.getInstance("SHA-256"); |
|  | byte[] hashInBytes = md.digest(str.getBytes(StandardCharsets.UTF\_8)); |
|  | StringBuilder sb = new StringBuilder(); |
|  | for (byte b : hashInBytes) { |
|  | sb.append(String.format("%02x", b)); |
|  | } |
|  | return sb.toString(); |
|  | } |
|  | } |

|  |
| --- |
| import java.util.\*; |
|  |  |
|  |  |
|  | public class Task9 { |
|  | public static void main(String[] args) { |
|  | Scanner input = new Scanner(System.in); |
|  | System.out.printf("Print in wrong title: "); |
|  | String title = input.nextLine(); |
|  | System.out.printf("Correct title: %s\n", correctTitle(title)); |
|  | } |
|  |  |
|  |  |
|  | public static String correctTitle(String title) { |
|  | String[] words = title.split(" "); |
|  | String[] constWords = new String[] {"in", "at", "of", "and"}; |
|  | String word; |
|  | String result = ""; |
|  | FIRST: |
|  | for (int i = 0; i < words.length; i++) { |
|  | word = words[i]; |
|  | for (String constWord: constWords) { |
|  | if (constWord.equals(word.toLowerCase())) { |
|  | result += constWord + " "; |
|  | continue FIRST; |
|  | } |
|  | } |
|  | result += String.valueOf(word.charAt(0)).toUpperCase() + word.substring(1).toLowerCase() + " "; |
|  | } |
|  | return result.trim(); |
|  | } |
|  | } |

|  |
| --- |
| import java.util.\*; |
|  |  |
|  |  |
|  | public class Task10 { |
|  | public static void main(String[] args) { |
|  | Scanner input = new Scanner(System.in); |
|  | System.out.printf("Print in nubmer: "); |
|  | int number = Integer.parseInt(input.nextLine()); |
|  | System.out.println("Result: "); |
|  | System.out.println(hexLattice(number)); |
|  | } |
|  |  |
|  |  |
|  | public static String hexLattice(int number) { |
|  | if (number % 6 != 1) return "Invalid"; |
|  | if (number == 1) return "o"; |
|  | int levels = 1; |
|  | int i = 0; |
|  | String result = ""; |
|  | while (levels != number) { |
|  | levels += 6 \* i; |
|  | i++; |
|  | } |
|  | for (int k = i; k <= 2 \* i - 1; k++) { |
|  | for (int j = k; j < 2 \* i - 1; j++) { |
|  | result += " "; |
|  | } |
|  | for (int j = 0; j < k; j++) { |
|  | result += "o "; |
|  | } |
|  | result += "\n"; |
|  | } |
|  | for (int k = 2 \* i - 2; k >= i; k--) { |
|  | for (int j = k; j < 2 \* i - 1; j++) { |
|  | result += " "; |
|  | } |
|  | for (int j = 0; j < k; j++) { |
|  | result += "o "; |
|  | } |
|  | result += "\n"; |
|  | } |
|  | return String.valueOf(result); |
|  | } |
|  | } |