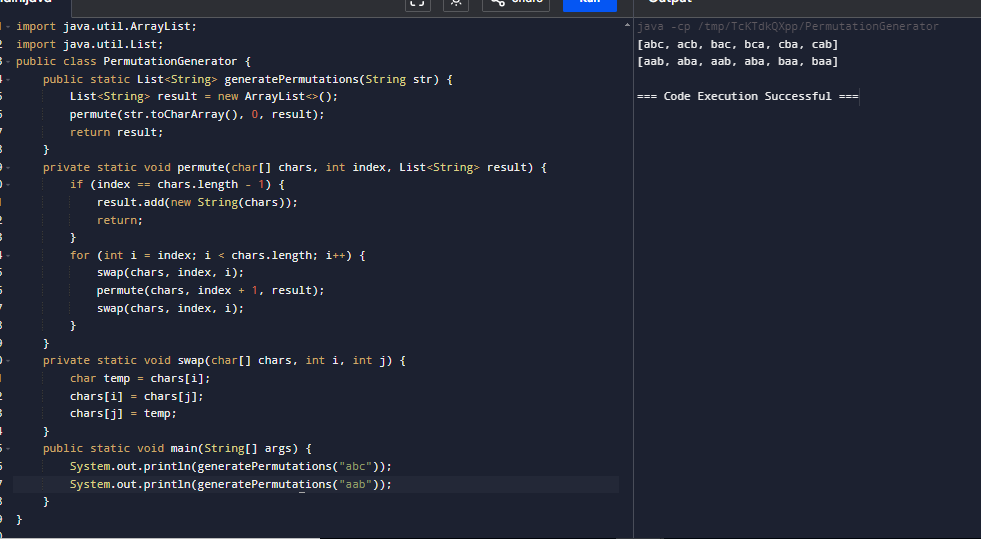
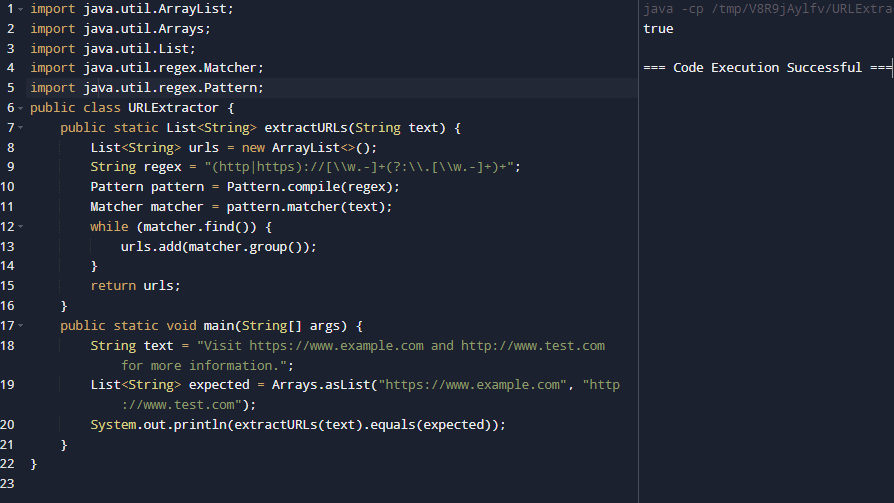
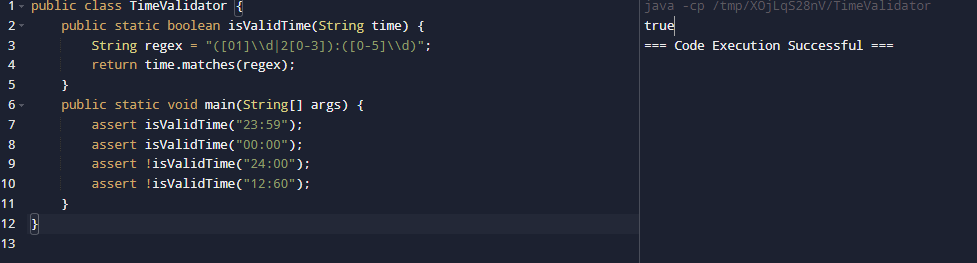
Java ***JAVA TEST-9(28-08-2024)est-***9 (28-08-2024)

1.Write a recursive method to generate all permutations of a given string. Test Cases: "abc" → [abc, bac, bca, cab, cba] "aab" → [aab, aba, baa]

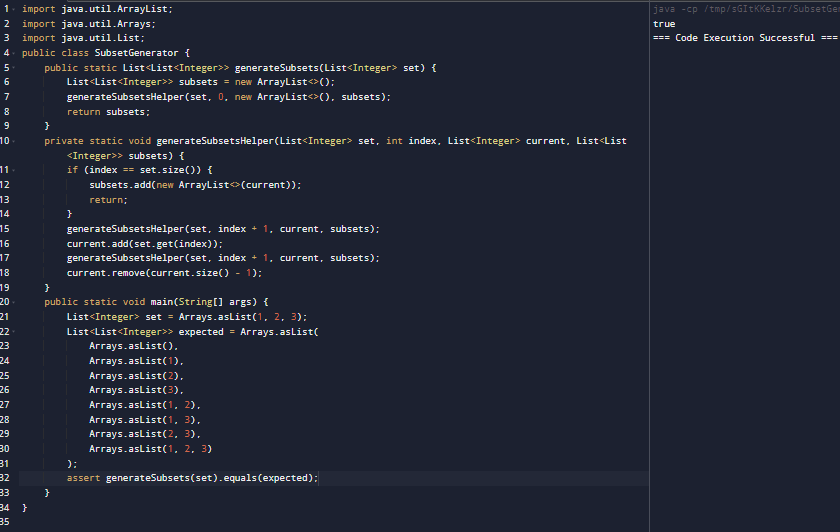


**2**.Write a Java method that extracts all valid URLs from a given string. A valid URL must start with http or https, followed by ://, and should contain a valid domain name. Test Cases: String text = "Visit https://www.example.com and http://www.test.com for more information."; List expected = Arrays.asList("https://www.example.com", "http://www.test.com"); assert extra

**3**.Write a Java method to validate if a given string represents a valid time in the 24-hour format (HH). Test Cases: assert isValidTime("23:59"); // true assert isValidTime("00:00"); // true assert !isValidTime("24:00"); // false assert !isValidTime("12:60"); // false



**4**.Write a recursive Java method to generate all subsets of a given set of integers. Test Cases: List set = Arrays.asList(1, 2, 3); List<List> expected = Arrays.asList( Arrays.asList(), Arrays.asList(1), Arrays.asList(2), Arrays.asList(3), Arrays.asList(1, 2), Arrays.asList(1, 3), Arrays.asList(2, 3), Arrays.asList(1, 2, 3) ); assert generateSubsets(set).equals(expected); // true

****

**5**.Write a recursive Java method to determine if a string can be segmented into a space-separated sequence of one or more dictionary words. Test Cases: Set wordDict = new HashSet<>(Arrays.asList("apple", "pen", "applepen", "pine", "pineapple")); assert wordBreak("pineapplepenapple", wordDict) == true; // true assert wordBreak("catsandog", wordDict) == false; // true

