Java Loops and Collections

CSC 210 Practice Exercises

FizzBuzz Array

Write a method that takes in an integer `k` as argument and returns an array with strings as values, for every integer from zero to `k`. The string should be `"FizzBuzz"` if `n` is divisible by 3 and 5, `"Fizz"` if `n` is divisible by 3, and `"Buzz"` if `n` is divisible by 5 or `n` (as a string) if none of the conditions are true.

Prime

Write a method that returns `true` if an integer `n` is prime, `false` otherwise.

You can determine if a number n is NOT prime if it is divisible by any number between 2 and n / 2.

The integer 1 is not a prime number.

Fibonacci Sequence

Fibonacci numbers are a sequence of numbers where every number is the sum of the preceding two numbers.

Write a method that takes in an integer `k` as argument and returns an array with integer as values, representing a Fibonacci sequence of size `k`. Assume `k >= 2`.

Prime numbers

Write a method that returns an array with the first `n` prime numbers.

Count Vowels

Write a method that takes as argument a word (string) and returns a `HashMap` with the vowel counts for the word.

Test cases:

- for `"banana"` the method returns `{'a'=3, 'e'=0, 'u'=0, 'i'=0, 'o'=0}`
- for `"sequoia"` the method returns `{'a'=1, 'e'=1, 'u'=1, 'i'=1, 'o'=1}`

Count Character Type

Write a method that takes as argument a word (string) and returns a `HashMap` with the how many vowels and consonants the word has.

Test cases:

- for `"banana"` the method returns `{"consonant"=3, "vowel"=3}`
- for `"sequoia"` the method returns `{"consonant"=2, "vowel"=5}`

ANSWERS

FizzBuzz Array

```
public static String fizzBuzzSingle(int n) {
        String result = "";

        if (n % 3 == 0) result += "Fizz";
        if (n % 5 == 0) result += "Buzz";

        if (result.equals("")) result += n;

        return result;
    }

public static ArrayList<String> fizzBuzz(int n) {
        ArrayList<String> result = new ArrayList<String>();

        for (int i = 0; i <= n; i++) {
            result.add(fizzBuzzSingle(i));
        }

        return result;

    }
```

Prime

```
public static boolean isPrime(int n) {

        if (n == 1) return false;

        for (int i = 2; i <= n/2; i++) {
            if (n % i == 0) return false;
        }

        return true;
    }
```

Fibonacci Sequence

```
public static ArrayList<Integer> fibonacci(int k) {
        ArrayList<Integer> result = new ArrayList<Integer>();

        result.add(0);
        result.add(1);

        for (int i = 2; i < k; i++) {
            result.add(result.get(i-1) + result.get(i-2));
        }

        return result;
    }
```

Prime numbers

```
public static boolean isPrime(int n) {

        if (n == 1) return false;

        for (int i = 2; i <= n/2; i++) {
            if (n % i == 0) return false;
        }

        return true;
    }

public static ArrayList<Integer> firstPrimes(int n) {
        ArrayList<Integer> result = new ArrayList<Integer>();

        int current = 1;
        while (result.size() < n) {
            if (isPrime(current)) result.add(current);
            current += 1;
        }

        return result;
    }
```

Count Vowels

```java
public static HashMap<Character, Integer> countVowels(String word) {
        HashMap<Character, Integer> result = new HashMap<Character, Integer>();
        result.put('a', 0);
        result.put('e', 0);
        result.put('i', 0);
        result.put('o', 0);
        result.put('u', 0);

        for (char c : word.toCharArray()) {
            if (result.containsKey(c)) result.put(c, result.get(c) + 1);
        }
        return result;
    }
```

Count Character Type

```java
public static HashMap<String, Integer> countCharType(String word) {
        HashMap<String, Integer> result = new HashMap<String, Integer>();
        result.put("vowel", 0);
        result.put("consonant", 0);

        HashSet<Character> vowels = new HashSet<Character>();
        vowels.add('a'); vowels.add('e'); vowels.add('i');
        vowels.add('o'); vowels.add('u');

        for (char c : word.toCharArray()) {
            if (vowels.contains(c)) result.put("vowel", result.get("vowel") + 1);
            else result.put("consonant", result.get("consonant") + 1);

        }

        return result;
    }
```