Ромашкина Анастасия

**Отчет по лабораторной работе №13**

**Задание №4**

* **toLowerCase(): String**

возвращает копию строки, преобразованной в нижний регистр.

* **toUpperCase(): String**

возвращает копию строки, преобразованной в верхний регистр.

* **substring():String**

возвращает подстроку, заданную определенным диапазоном.

* **reversed(startIndex: Int, endIndex: Int): String**

возвращает подстроку этой строки, начинающуюся с startIndex и заканчивающуюся непосредственно перед endIndex.

* **replaceFirst(regex: Regex, replacement: String): String**

заменяет первое вхождение данного регулярного выражения regex в этой последовательности символов указанным выражением.

* **replace(regex: Regex, replace: String): String**

возвращает новую строку, полученную путем замены каждой подстроки этой последовательности символов, которая соответствует данному регулярному выражению.

* **trim(predicate: (Char) -> Boolean): String**

возвращает строку, содержащую начальные и конечные символы, соответствующие удаленному предикату.

* **take(n: Int): String**

возвращает строку, содержащую первые n символов из этой строки, или всю строку, если эта строка короче.

* **removeRange(startIndex: Int, endIndex: Int): String**

удаляет часть строки в заданном диапазоне.

* **length: Int**

возвращает длину строки.

* **compareTo(other: String): Int**

сравнивает строку other. Возвращает ноль, если строка равна other; отрицательное число, если оно меньше other; или положительное число, если оно больше other.

* **equals(other: Any?): Boolean**

определяет, равна ли строка other.

* **get(index: Int): Char**

возвращает символ строки с указанным номером.

* **hashCode(): Int**

возвращает хэш-код строки.

* **codePointAt(index: Int): Int**

возвращает код символа Unicode.

* **count(): Int**

возвращает количество символов.

* **drop(n: Int): String**

возвращает строку с первыми удаленными n символами.

* **find(predicate: (Char) -> Boolean): Char?**

возвращает первый символ, соответствующий данному предикату, или null, если такой символ не был найден.

* **isEmpty(): Boolean**

возвращает true, если последовательность символов пуста.

**Задание №6**

* **and(other: Int): Int**

выполняет побитовую операцию «И» между двумя значениями.

* **compareTo(other: Int): Int**

сравнивает значение с other. Возвращает ноль, если это значение равно other, отрицательное число, если оно меньше other, или положительное число, если оно больше other.

* **div(other: Int): Int**

делит значение на other, усекая результат до целого числа, которое ближе к нулю.

* **equals(other: Int): Boolean**

определяет, равно ли число other.

* **hashCode(): Int**

возвращает хэш-код числа.

* **inv(): Int**

инвертирует биты числа.

* **or(other: Int): Int**

выполняет побитовую операцию «ИЛИ» между двумя значениями.

* **rem(other: Int): Int**

вычисляет остаток от усечения деления этого значения на other.

* **shl(bitCount: Int): Int**

смещает значение влево на количество битов bitCount.

* **shr(bitCount: Int): Int**

смещает значение вправо на количество битов bitCount, заполняя крайние левые биты копиями знакового бита.

* **times(other: Int): Int**

умножает значение на other.

* **xor(other: Int): Int**

выполняет побитовую операцию «XOR» между двумя значениями.

* **mod(other: Int): Int**

вычисляет остаток от деления этого значения на другое значение.