Общее

val *isMoreOrEqual*: (Int, Int) -> Boolean = **{** num, num1 **->** num >= num1 **}**val *isLessOrEqual*: (Int, Int) -> Boolean = **{** num, num1 **->** num <= num1 **}**fun main(args: Array<String>) {  
 val str = "111111 22 3333 4 555 6666 77 88888"  
 if (str.*isEmpty*())  
 {  
 *println*("Ответ: Строка пустая")  
 return  
 }  
 val Condition: (Int, Int) -> Boolean = *isLessOrEqual* var res = *findLastChar*(  
 str  
 ,**{it**.*isWhitespace*() || **it** == '.'**}** ,**{it** % 2 == 0**}** ,Condition  
 )  
 if (res == ' ') {  
 *println*("Нет ответа")  
 } else {  
 val wordType = if (Condition == *isMoreOrEqual*) "самом длинном" else "самом коротком"  
 *println*("Последний символ в последнем $wordType слове\nс чётным числом символов = $res")  
 }  
  
}  
  
  
fun findLastChar(  
 str: String,  
 isNotSymbol: (Char) -> Boolean = **{ it**.*isWhitespace*() || **it** == '.' **}**,  
 isEven: (Int) -> Boolean = **{ it** % 2 == 0 **}**,  
 condition: (Int, Int) -> Boolean = **{** num, num1 **->** num <= num1 **}**): Char {  
 var size = 0  
 var maxSize = 0  
 var result = ' '  
  
 if(condition(size, maxSize) == *isMoreOrEqual*(size, maxSize))  
 {  
 for (i in str.*indices*) {  
 if (isNotSymbol(str[i])) {  
 if (isEven(size) && size > 0 && condition(size, maxSize)) {  
 result = str[i - 1]  
 maxSize = size  
 }  
 size = 0  
 } else {  
 size++  
 }  
 }  
  
 // Проверяем последнее слово после цикла  
 if (isEven(size) && size > 0 && condition(size, maxSize))  
 {  
 // Добавим проверку на четность длины последнего слова  
 val lastWordLength = if (size == maxSize) size else size + 1 // Длина последнего слова  
 if (isEven(lastWordLength)) {  
 result = if (size == maxSize) str.*last*() else str[str.length - size] // Учтем, если последнее слово также самое длинное  
 }  
 }  
 }  
 else if(condition(size, maxSize) == *isLessOrEqual*(size, maxSize)) //ошибка  
 {  
 for (i in 0 *until* str.length)  
 {  
 if (isNotSymbol(str[i]))  
 {  
 if (isEven(size) && condition(size, maxSize)) //\*обработать >=  
 {  
 result = str[i - 1]  
 maxSize = size  
 }  
 size = 0  
 } else  
 size++  
 }  
 if (isEven(size) && condition(size, maxSize)) {  
 result = str[str.length - 1]  
 maxSize = size  
 }  
 }  
 return result  
}

длинное

fun main(args: Array<String>) {  
 val str = "111111 22 3333 4 555 6666 77 88888 999999"  
 if (str.*isEmpty*()) {  
 *println*("Ответ: Строка пустая")  
 return  
 }  
  
 // Определение условия отбора  
 val isMoreOrEqual: (Int, Int) -> Boolean = **{** num, num1 **->** num >= num1 **}** val isLessOrEqual: (Int, Int) -> Boolean = **{** num, num1 **->** num <= num1 **}** // Выбор условия  
 val condition: (Int, Int) -> Boolean = isMoreOrEqual  
  
  
 val result = *findLastChar*(  
 str,  
 **{ it**.*isWhitespace*() || **it** == '.' **}**,  
 **{** num **->** num % 2 == 0 **}**,  
 condition  
 )  
  
 if (result == ' ') {  
 *println*("Нет ответа")  
 } else {  
 val wordType = if (condition == isMoreOrEqual) "самом длинном" else "самом коротком"  
 *println*("Последний символ в последнем $wordType слове\nс чётным числом символов = $result")  
 }  
}  
  
fun findLastChar(  
 str: String,  
 isNotSymbol: (Char) -> Boolean = **{ it**.*isWhitespace*() || **it** == '.' **}**,  
 isEven: (Int) -> Boolean = **{ it** % 2 == 0 **}**,  
 condition: (Int, Int) -> Boolean = **{** num, num1 **->** num >= num1 **}**): Char {  
 var size = 0  
 var maxSize = 0  
 var result = ' '  
  
 for (i in str.*indices*) {  
 if (isNotSymbol(str[i])) {  
 if (isEven(size) && size > 0 && condition(size, maxSize)) {  
 result = str[i - 1]  
 maxSize = size  
 }  
 size = 0  
 } else {  
 size++  
 }  
 }  
  
 // Проверяем последнее слово после цикла  
 if (isEven(size) && size > 0 && condition(size, maxSize))  
 {  
 // Добавим проверку на четность длины последнего слова  
 val lastWordLength = if (size == maxSize) size else size + 1 // Длина последнего слова  
 if (isEven(lastWordLength)) {  
 result = if (size == maxSize) str.*last*() else str[str.length - size] // Учтем, если последнее слово также самое длинное  
 }  
 }  
 return result  
}

Короткое

f fun main(args: Array<String>) {  
 val str = "111111 22 3333 4 555 6666 77 88888"  
 if (str.isEmpty())  
 {  
 println("Ответ: Строка пустая")  
 return  
 }  
 var res = findLastChar(  
 str  
 ,**{**it.isWhitespace() || it == '.'**}** ,**{**it % 2 == 0**}** )  
 if(res == ' ')  
 {  
 println("Нет ответа")  
 return  
 }  
 else  
 println("Последний символ в последнем самом коротком слове\n" +  
 " с чётным числом символов: ${res}") //чётное слов  
}  
fun findLastChar(str: String  
 ,IsNotSymbol:(Char) -> Boolean = **{it**.*isWhitespace*() || **it** == '.'**}** ,IsEven: (Int) -> Boolean = **{it** % 2 == 0**}**) : Char  
{  
 var size = 0  
 var oldSize = Int.MAX\_VALUE  
 var res = ' '  
 for (i in 0 *until* str.length)  
 {  
 if (IsNotSymbol(str[i]))  
 {  
 if (IsEven(size) && size <= oldSize)   
 {  
 res = str[i - 1]  
 oldSize = size  
 }  
  
 size = 0  
 }  
 else  
 size++  
 }  
 if (IsEven(size) && size <= oldSize)  
 {  
 res = str[str.length - 1]  
 oldSize = size  
 }  
 return res  
}

val *isMoreOrEqual*: (Int, Int) -> Boolean = **{** num, num1 **->** num >= num1 **}**val *isLessOrEqual*: (Int, Int) -> Boolean = **{** num, num1 **->** num <= num1 **}**fun main(args: Array<String>) {  
 val str = "111111 22 3333 4 555 6666 77 88888"  
 if (str.*isEmpty*())  
 {  
 *println*("Ответ: Строка пустая")  
 return  
 }  
 val Condition: (Int, Int) -> Boolean = *isLessOrEqual* var res = *findLastChar*(  
 str  
 ,**{it**.*isWhitespace*() || **it** == '.'**}** ,**{it** % 2 == 0**}** ,Condition  
 )  
 if (res == ' ') {  
 *println*("Нет ответа")  
 } else {  
 val wordType = if (Condition == *isMoreOrEqual*) "самом длинном" else "самом коротком"  
 *println*("Последний символ в последнем $wordType слове\nс чётным числом символов = $res")  
 }  
  
}  
  
  
fun findLastChar(  
 str: String,  
 isNotSymbol: (Char) -> Boolean = **{ it**.*isWhitespace*() || **it** == '.' **}**,  
 isEven: (Int) -> Boolean = **{ it** % 2 == 0 **}**,  
 condition: (Int, Int) -> Boolean = **{** num, num1 **->** num <= num1 **}**): Char {  
 var size = 0  
 var maxSize = 0  
 var result = ' '  
  
 if(condition(size, maxSize) == *isMoreOrEqual*(size, maxSize))  
 {  
 for (i in str.*indices*) {  
 if (isNotSymbol(str[i])) {  
 if (isEven(size) && size > 0 && condition(size, maxSize)) {  
 result = str[i - 1]  
 maxSize = size  
 }  
 size = 0  
 } else {  
 size++  
 }  
 }  
  
 // Проверяем последнее слово после цикла  
 if (isEven(size) && size > 0 && condition(size, maxSize))  
 {  
 // Добавим проверку на четность длины последнего слова  
 val lastWordLength = if (size == maxSize) size else size + 1 // Длина последнего слова  
 if (isEven(lastWordLength)) {  
 result = if (size == maxSize) str.*last*() else str[str.length - size] // Учтем, если последнее слово также самое длинное  
 }  
 }  
 }  
 else if(condition(size, maxSize) == *isLessOrEqual*(size, maxSize)) //ошибка  
 {  
 for (i in 0 *until* str.length)  
 {  
 if (isNotSymbol(str[i]))  
 {  
 if (isEven(size) && size <= maxSize) //\*обработать >=  
 {  
 result = str[i - 1]  
 maxSize = size  
 }  
  
 size = 0  
 }  
 else  
 size++  
 }  
 if (isEven(size) && size <= maxSize)  
 {  
 result = str[str.length - 1]  
 maxSize = size  
 }  
 }  
 return result  
}

val *isMoreOrEqual*: (Int, Int) -> Boolean = **{** num, num1 **->** num >= num1 **}**val *isLessOrEqual*: (Int, Int) -> Boolean = **{** num, num1 **->** num <= num1 **}**var *size* = 0  
var *oldSize* = Int.MAX\_VALUE  
  
fun main(args: Array<String>) {  
 val str = "111111 22 3333 4 555 6666 77 88888"  
 if (str.*isEmpty*()) {  
 *println*("Ответ: Строка пустая")  
 return  
 }  
 val condition: (Int, Int) -> Boolean = *isMoreOrEqual* var res = *findLastChar*(  
 str,  
 **{ it**.*isWhitespace*() || **it** == '.' **}**,  
 **{ it** % 2 == 0 **}**,  
 condition  
 )  
 if (res == ' ') {  
 *println*("Нет ответа")  
 } else {  
 val wordType = if (condition(*size*, *oldSize*) == *isMoreOrEqual*(*size*, *oldSize*)) "самом длинном" else "самом коротком"  
 *println*("Последний символ в последнем $wordType слове\nс чётным числом символов = $res")  
 }  
}  
  
fun findLastChar(  
 str: String,  
 isNotSymbol: (Char) -> Boolean = **{ it**.*isWhitespace*() || **it** == '.' **}**,  
 isEven: (Int) -> Boolean = **{ it** % 2 == 0 **}**,  
 condition: (Int, Int) -> Boolean = **{** num, num1 **->** num <= num1 **}**): Char {  
 var result = ' '  
 if (condition(*size*, *oldSize*) == *isLessOrEqual*(*size*, *oldSize*))  
 {  
 for (i in 0 *until* str.length) {  
 if (isNotSymbol(str[i])) {  
 if (isEven(*size*) && condition(*size*, *oldSize*)) {  
 result = str[i - 1]  
 *oldSize* = *size* }  
 *size* = 0  
 } else  
 *size*++  
 }  
 if (isEven(*size*) && condition(*size*, *oldSize*))  
 {  
 result = str[str.length - 1]  
 *oldSize* = *size* }  
 }  
 else if (condition(*size*, *oldSize*) == *isMoreOrEqual*(*size*, *oldSize*))  
 {  
 for (i in str.*indices*) {  
 if (isNotSymbol(str[i])) {  
 if (isEven(*size*) && *size* > 0 && condition(*size*, *oldSize*)) {  
 result = str[i - 1]  
 *oldSize* = *size* }  
 *size* = 0  
 } else  
 *size*++  
 }  
 if (isEven(*size*) && *size* > 0 && condition(*size*, *oldSize*)) {  
 result = if (*size* == *oldSize*) str.*last*() else str[str.length - *size*]  
 }  
 }  
 return result  
}