Министерство образования Республики Беларусь

Учреждение высшего образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

ОТЧЕТ

по лабораторной работе №2

на тему

**Работа со списками и функциями**

|  |  |
| --- | --- |
| Студенты группы 150504  Проверила | Шевцов В.С  Крук А.И  Герман Ю.О. |

Минск 2023

1. **Цель работы**

Изучить технику работы со списками и функциями в Scala.

1. **Краткие теоретические сведения**

Примеры функций для работы со списками в Scala:

1. Map (карта,отображение) : эта функция применяет заданную функцию к каждому элементу списка и возвращает новый список с результатами.

object Main22 {

def double(x: Int): Int = x \* 2

def main(args: Array[String]): Unit = {

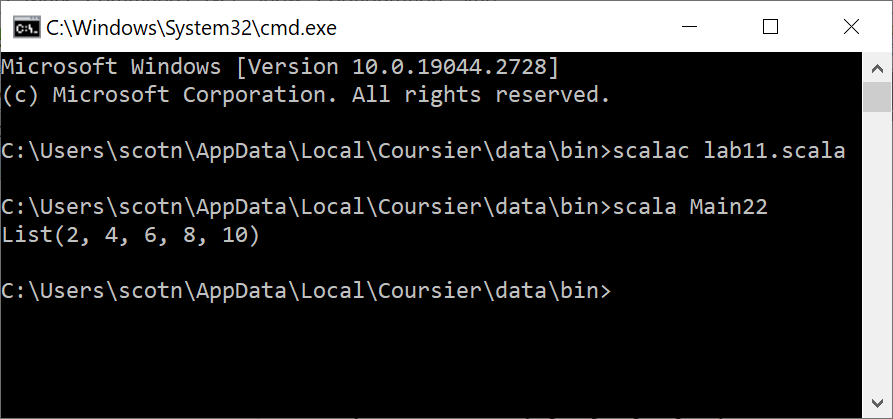
val myList = List(1, 2, 3, 4, 5)

val doubledList = myList.map(double)

println(doubledList) // Output: List(2, 4, 6, 8, 10)

}

}



Поэлементный вывод списка

object Main22 {

def double(x: Int): Int = x \* 2

def main(args: Array[String]): Unit = {

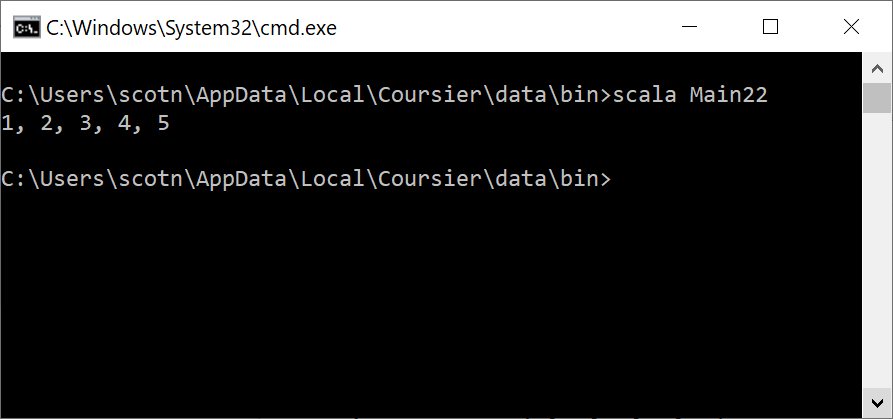
val myList = List(1, 2, 3, 4, 5)

val doubledList = myList.map(double)

println(myList.mkString(", "))

}

}



2. filter: Эта функция отбирает элементы списка, удовлетворяющие заданному предикату.

Следующий пример показывает, как вывести четные элементы списка

object Main22 {

def isEven(x: Int): Boolean = x % 2 == 0

def main(args: Array[String]): Unit = {

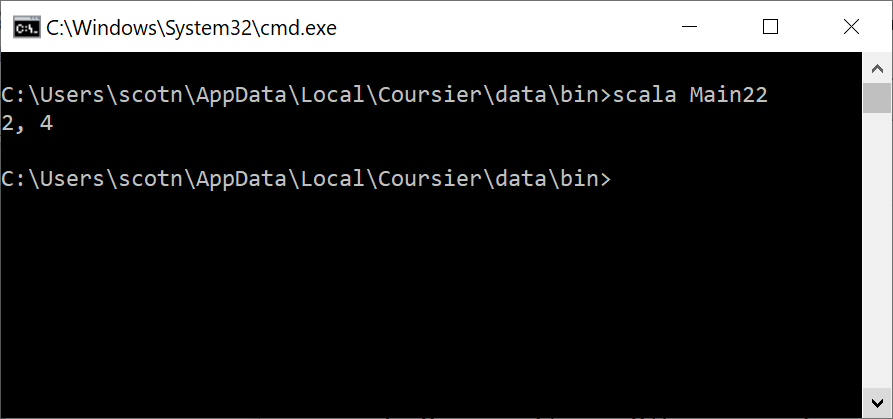
val myList = List(1, 2, 3, 4, 5)

val filteredList = myList.filter(isEven)

println(filteredList.mkString(", ")) // Output: List(2, 4)

}

}



3. foldLeft: Эта функция последовательно применяется к элементам списка слева направо, накапливая результат. Сумму элементов списка можно найти таким образом

object Main22 {

def main(args: Array[String]): Unit = {

val myList = List(1, 2, 3, 4, 5)

val sum = myList.foldLeft(0)((ac\_c, x) => ac\_c + x)

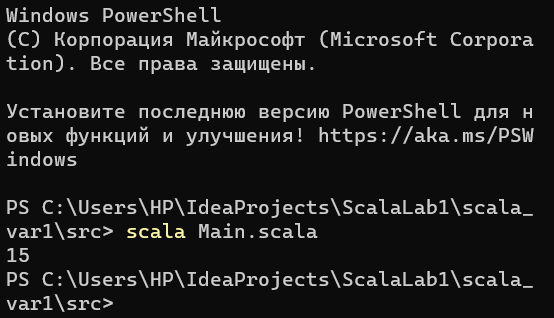
println(sum) // Output: 15

}

}

Здесь переменная ac\_c играет роль аккумулятора. Первоначально ей присваивается значение 0:

myList.foldLeft(0)



4. zip: Эта функция объединяет два списка на примере словаря (dictionary) – ключ-значение.

object Main22 {

def main(args: Array[String]): Unit = {

val a = List(1, 2, 3)

val b = List("one", "two", "three")

val zipped = a.zip(b) // List((1, "one"), (2, "two"), (3, "three"))

println(zipped)

val filteredList = zipped.filter { case (a, \_) => a == 1 }

println(filteredList)

val tup=filteredList.head

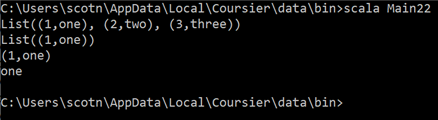
println(tup)

val secondItem = tup.\_2

println(secondItem)

}

}

****

5. head and tail: Эти функции возвращают голову и хвост списка соответственно.

Сумма квадратов элементов списка:

object Main22 {

def sumList(lst: List[Int]): Int = {

if (lst.isEmpty) 0

else lst.head\*lst.head + sumList(lst.tail)

}

def main(args: Array[String]): Unit = {

val myList = List(1, 2, 3, 4, 5)

val sumw = sumList(myList)

println(sumw)

}

}

Этот же пример можно реализовать по-другому

object Main22 {

def main(args: Array[String]): Unit = {

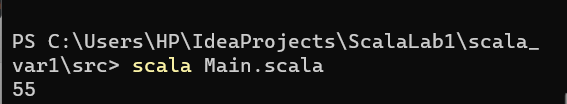
val myList = List(1, 2, 3, 4, 5)

val sumOfSquares = myList.map(item => item \* item).sum

println(sumOfSquares)

}

}



Подсчет числа отрицательных элементов списка (почти предыдущий вариант)

object Main22 {

def main(args: Array[String]): Unit = {

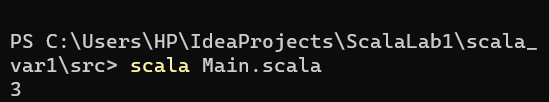
val myList = List(1, -2, 3, -7, -5)

val countNegative = myList.count(item => item < 0)

println(countNegative)

}

}



Отыскание минимального элемента в списке

object Main22 {

def minList(lst: List[Int], minval: Int): Int = {

if (lst.isEmpty) minval

else if (lst.head<minval) minList(lst.tail,lst.head)

else minList(lst.tail,minval)

}

def main(args: Array[String]): Unit = {

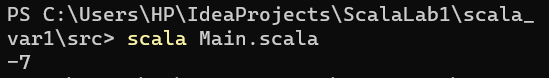
val myList = List(1, -2, 3, -7, -5)

val minw = minList(myList,myList.head)

println(minw)

}

}



Или для ввода списка с клавиатуры:

import scala.io.StdIn

object Main224 {

def minList(lst: List[Int], minval: Int): Int = {

if (lst.isEmpty) minval

else if (lst.head<minval) minList(lst.tail,lst.head)

else minList(lst.tail,minval)

}

def main(args: Array[String]): Unit = {

println("Enter elements of the list (integers), separated by spaces:")

val myList = StdIn.readLine().split(" ").map(\_.toInt).toList

println("Entered list: " + myList)

if (myList.isEmpty) {

println("The list is empty")

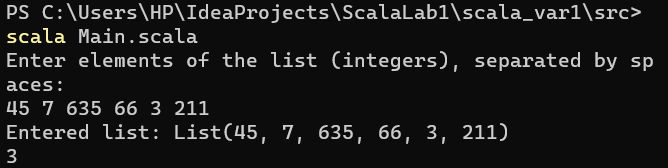
} else

val minw = minList(myList,myList.head)

println(minw)

}

}

****

Отыскание элемента на позиции *i* при нумерации позиций с нуля, список содержит положительные целые числа. Если индекс выходит за пределы списка, возвращается -1:

object Main22 {

def posList(lst: List[Int], pos: Int): Int = {

if (lst.isEmpty) -1

else if (pos==0) lst.head

else {val newpos=pos-1

posList(lst.tail,newpos)}

}

def main(args: Array[String]): Unit = {

val myList = List(1, 2, 3, 4, 5)

val pos =3

val minw = posList(myList,pos)

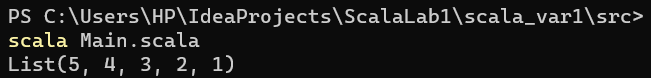
println(minw)

}

}

****

6. reverse: Эта функция возвращает список в обратном порядке.

****

**Задание**

1. Написать функцию для подсчета суммы элементов списка, значение которых по модулю не превосходит 5. Список задать самостоятельно.
2. Написать функцию для подсчета суммы каждого второго элемента списка из 10 элементов. Список задать самостоятельно.
3. Написать функцию для отыскания индекса элемента списка, наименее отклоняющегося от среднего значения по списку. Список задать самостоятельно.
4. Написать функцию для проверки того, что список содержит квадрат одного из своих элементов. Список задать самостоятельно.
5. Написать функцию для проверки наличия трех разных элементов в списке. Список задать самостоятельно. Функция возвращает значение Да или Нет

**Исходный код**

№1

object Main {

def main(args: Array[String]): Unit = {

val list = List(-1, 0, 1, -2, 2, 3, 4, 6, 10 ,5)

val sum = list.filter(num => abs(num) <= 5).sum

println(sum)

}

}



№2

object Main2 {

private def sum(list: List[Int], i: Int): Int = {

var accum = 0;

if (i == list.size)

return accum

if (i % 2 != 0)

accum = list.apply(i)

accum += sum(list, i + 1)

accum

}

def main(args: Array[String]) = {

val list = List(-1, 0, 1, -2, 2, 3, 4, 6, 10, 5)

val accum = sum(list, 0)

println(accum)

}

}



№3

object Main3 {

private def middle(list: List[Int], criteria: Float): Int = {

var dif: Float = abs(list.head - criteria)

var elem = 0

for (i <- list.indices if abs(list.apply(i) - criteria) < dif)

{

elem = list.apply(i)

dif = abs(list.apply(i) - criteria)

}

elem

}

def main(args: Array[String]) = {

val list = List(-1, 0, 1, -2, 2, 3, 4, 6, 10, 5)

val sum = list.sum.toFloat / list.size

var elem = middle(list, sum)

println(elem)

}

}



№4

@scala.annotation.nowarn

object Main {

def main(args: Array[String]): Unit = {

val list = List(-1, 0, 1, -2, 2, 3, 4, 6, 10, 5)

for (i <- 0 until list.size)

for (j <- i + 1 until list.size)

if (scala.math.pow(i.toDouble, 2) == list.apply(j)) {

println("List contains square of one of its elements")

return

}

println("List not contains square of one of its elements")

}

}



№5

object Main5 {

private def collisions(list: List[Int], element: Int, counter: Int): Int = {

var counter2 = 0

if (list.isEmpty)

return counter

if (list.contains(element))

counter2 = collisions(list.tail, list.head, counter + 1)

else

counter2 = collisions(list.tail, list.head, counter)

counter2

}

def main(args: Array[String]) = {

var counter = 0

val list = List(-1, 0, 1, -2, 2, 3, 4, 6, 5, 5)

counter = collisions(list.tail, list.head, 0)

if (counter > 6)

println("No")

else

println("Yes")

}

}

