Упражнения:

1. Напишите фрагмент кода, который записывает в массив а целые числа в диапазоне от 0 (включительно) до n (исключая его).

**def** enterNumberMas(a: ArrayBuffer[Int],n:Int): Unit ={  
 a ++= 0 until n  
}



1. Напишите цикл, меняющий местами смежные элементы в массиве целых чисел. Например, Аггау(1, 2, 3, 4, 5) должен стать   
   Аггау(2, 1, 4, 3, 5).

**def** switchNumberMas(a: ArrayBuffer[Int]): ArrayBuffer[Int] ={  
 **for** (i <-1 until a.length **if** i%2==1){  
 **val** b=a(i-1)  
 a(i-1)=a(i)  
 a(i)=b  
 }  
 a  
}

 

1. Повторите предыдущее упражнение, но создайте новый массив с переставленными элементами. Используйте выражение for/yield.

**def** switchNumberMasYield(a: ArrayBuffer[Int]) ={  
 **var** b = ArrayBuffer[Int]()  
 b++=(**for** (i <-0 until a.length)**yield**{  
 **if** (i==a.length-1){  
 **if**(i%2==1){  
 a(i-1)  
 }  
 **else**{  
 a(i)  
 }  
 }  
 **else**{  
 **if**(i%2==0){  
 a(i+1)  
 }  
 **else** a(i-1)  
 }  
 })  
 b  
}

 

1. Дан массив целых чисел, создайте новый массив, в котором сначала будут следовать положительные значения из оригинального массива, в оригинальном порядке, а за ними отрицательные и нулевые значения, тоже в оригинальном порядке.

**def** sortZero(a: ArrayBuffer[Int]): ArrayBuffer[Int] ={  
 **var** b = ArrayBuffer[Int]()  
 b = **for** (elem <- a **if** elem > 0) **yield** elem  
 b++=(**for** (elem <- a **if** elem <= 0) **yield** elem)  
}



1. Как бы вы вычислили среднее значение элементов массива Аггау[Double]?

**def** averageMas(a: ArrayBuffer[Double]): Double ={  
 a.sum/a.size  
}



1. Как бы вы переупорядочили элементы массива Аггау[Int] так, чтобы они следовали в обратном отсортированном порядке? Как бы вы сделали то же самое с буфером ArrayBuffer[Int]?

**def** reverseSortMas(a: ArrayBuffer[Int]): ArrayBuffer[Int] ={  
 a.sortWith(\_ > \_)  
}



1. Напишите фрагмент программного кода, выводящий значения всех элементов из массива, кроме повторяющихся. (Подсказка: загляните в Scaladoc.)

**def** distinctMas(a: ArrayBuffer[Int]): ArrayBuffer[Int] ={  
 a.distinct  
}



1. Перепишите пример в конце раздела 3.4 «Преобразование массивов». Соберите индексы отрицательных элементов, расположите их в обратном порядке, отбросьте последний индекс и вызовите а.remove(1) для каждого индекса. Сравните эффективность этого подхода с двумя подходами, представленными в разделе 3.4.

«Дана последовательность целых чисел, из которой требуется удалить все отрицательные числа, кроме первого»

**def** delNegativNumberMas(a: ArrayBuffer[Int]): ArrayBuffer[Int] = {  
 **var** b = ArrayBuffer[Int]()  
 b.++=(**for** (i <-0 until a.length **if** a(i)<0) **yield** i).trimStart(1)  
 b=b.reverse  
 **var** c = ArrayBuffer[Int]()  
 c++=(**for** (i <- 0 until a.length **if** !b.contains(i))**yield** a(i))  
}



1. Создайте коллекцию всех часовых поясов, возвращаемых методом java.util.TimeZone.getAvailableIDs для Америки. Отбросьте префикс “America/" и отсортируйте результат.

**def** timeZoneAmerica(): ArrayBuffer[String] ={  
 **var** a = java.util.TimeZone.*getAvailableIDs*()  
 **var** b =ArrayBuffer[String]()  
 b++=(**for** (elem <- a **if** elem.contains("America/"))**yield** elem.drop(8)).sorted  
}

1. Импортируйте java.awt.datatransfer.\_ и создайте объект типа SystemFlavorMap вызовом val flavors = SystemFlavorMap.  
   getDefaultFlavorMap().asInstanceOf[SystemFlavorMap]  
     
   Затем вызовите метод getNativesForFlavor с параметром DataFlavor.  
   imageFlavor и получите возвращаемое значение как буфер Scala.

**def** flavor(): Unit ={  
 **val** flavors = SystemFlavorMap.*getDefaultFlavorMap*().  
 asInstanceOf[SystemFlavorMap]  
 **var** a=flavors.getNativesForFlavor(DataFlavor.*imageFlavor*)  
 *print*(a)  
}

