Федеральное агентство связи

Ордена Трудового Красного Знамени

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский технический университет связи и информатики»

Кафедра МКиИТ

Отчет по лабораторной № 1
«Функциональное программирование»

по дисциплине

«Введение в ИТ»

Выполнил: студент группы

БВТ1904

Сушков И.А

Руководитель:

Мосева М.С.

Москва 2021

1. Переменные res это значения val.

2.

```
scala> "crazy" * 3
val res1: String = crazycrazycrazy
```

3. Метод возвращает большее из двух чисел. Определен в RichInt

4. Используя число типа BigInt, вычислите 2^1024

```
BigInt(2) pow 1024

scala> val res0: scala.math.BigInt =
17976931348623159077293051907890247336179769789423065727343008115773267580550096313270847732240753602112011387987139335765878976881441662249284743063947416
24377767893424865485276302219601246094119453082952085005768838150682342462881473913110540827237163350510684586298239947245938479716304835356329624224137216
```

5. Что нужно импортировать, чтобы найти случайное простое число вызовом метода probablePrime(100, Random) без использования каких-либо префиксов перед именами probablePrime и Random?

```
scala> import scala.util.Random
import scala.util.Random

scala> import scala.math.BigInt
import scala.math.BigInt

scala> import scala.math.BigInt.probablePrime
import scala.math.BigInt.probablePrime

scala> import scala.util.Random

 probablePrime(100, Random)
import scala.util.Random

scala>
scala> val res3: scala.math.BigInt = 71686043723659976712627394753 1
```

6. Один из способов создать файл или каталог со случайным именем состоит в том, чтобы сгенерировать случайное число типа BigInt и преобразовать его в систему счисления по основанию 36, в результате получится строка, такая как "qsnvbevtomcj38o06kul". Отыщите в Scaladoc методы, которые можно было бы использовать для этого.

```
scala> probablePrime(100, Random)
val res3: scala.math.BigInt = 674308129069121354092722992161
```

7. Как получить первый символ строки в языке Scala? А последний символ?

```
scala> "First".head
val res4: Char = F
```

8. Что делают строковые функции take, drop, takeRight и dropRight? Какие преимущества и недостатки они имеют в сравнении с substring?

drop: Выбирает все элементы кроме первых x элементов

dropRight: Выбирает все элементы кроме последних x элементов

take: Выбирает первые x элементов

takeRight: Выбирает последние x элементов

```
scala> "First".head
val res4: Char = F

scala> "Hello World!" take 5
val res5: String = Hello

scala> "Hello World!" drop 5
val res6: String = " World!"

scala> "Hello World!" takeRight 5
val res7: String = orld!

scala> "Hello World!" dropRight 5
val res8: String = Hello W
```

9. Сигнум числа равен 1, если число положительное. -1 – если отрицательное, и 0 – если равно нулю. Напишите функцию, вычисляющую это значение.

```
def signum(num:Int) = if (num > 0) 1 else if (num < 0) -1 else 0

scala> def signum(num: Int): Int

scala> signum(7)
val res0: Int = 1

scala> signum(0)
val res1: Int = 0

scala> signum(-7)
val res2: Int = -1
```

10. Какое значение возвращает блок {}? Каков его тип
Значение 'no value', тип - Unit

11. Напишите на языке Scala цикл, эквивалентный циклу на языке Java for (int i=10; i>=0; i--) System.out.println(i);

```
scala> for (counter <- 10 to 0 by -1){
   println(counter);
 }
     |       | 10
9
8
7
6
5
4
3
2
1
0
```

12. Напишите процедуру countdown (n: Int), которая выводит числа от n до 0

```
scala> def countdown(n:Int) = for(i <- n to 0 by -1) println(i)
def countdown(n: Int): Unit

scala> countdown(10)
10
9
8
7
6
5
4
3
2
1
0
```

13. Напишите цикл for для вычисления кодовых пунктов Юникода всех букв в строке. Например, произведение символов в строке «Hello» равно 9415087488L.

```
scala> var h: Long = 1
var h: Long = 1

scala> for(i <- "Hello"){
   h = h * i.toLong
 }
     |      |
scala> h
val res1: Long = 9415087488
```

14. Решите предыдущее упражнение без применения цикла. Напишите функцию product(s: String), вычисляющую произведение, как описано в предыдущих упражнениях.

```
scala> var hello:Long = 1
var hello: Long = 1

scala> "Hello".foreach(hello *= _.toLong)

scala> hello
val res1: Long = 9415087488
```

16. Сделайте функцию из предыдущего упражнения рекурсивной.

```
scala> def product(hello:String):Long={
   if(hello.length == 1) return hello.charAt(0).toLong
   else hello.take(1).charAt(0).toLong * product(hello.drop(1))
 }
     |      |      | def product(hello: String): Long

scala> product("Hello")
val res1: Long = 9415087488
```

17. Напишите функцию, вычисляющую xn, где n – целое число.

Используйте следующее рекурсивное определение:

• xn = y2, если n – четное и положительное число, где y=xn/2

• xn = x*xn-1, если n – нечетное и положительное число.

• x0=1.

• xn=1/x-n, если n – отрицательное число.

Не используйте инструкцию return.

```
def seventeen(x:Double,n:Int):Double={
   if(n == 0) 1
   else if(n>0 && n%2!=0) x * seventeen(x,n-1)
   else if(n>0 && n%2==0) seventeen(x,n)/2 * seventeen(x,n)/2
   else 1/seventeen(x,-n)
 }

scala>      |      |      |      |      | def seventeen(x: Double, n: Int): Double

scala> seventeen(2,0)
val res0: Double = 1.0
```

18. f(m,n) - сумма всех натуральных чисел от m до n включительно, в десятичной записи которых нет одинаковых цифр.

```
scala> def f(m: Int, n:Int): Int = (m to n).filter(x=> (x.toString.size == x.toString.toSet.size)).sum
def f(m: Int, n: Int): Int

scala> f(110, 119)
val res0: Int = 0

scala> f(1,9)
val res1: Int = 45

scala> f(1,10)
val res2: Int = 55

scala> f(1,100)
val res3: Int = 4455
```

19. Список содержит целые числа, а также другие списки, такие же как и первоначальный. Получить список, содержащий только целые числа из всех вложенных списков. Пример: f(List(List(1, 1), 2, List(3, List(5, 8)))) = List(1, 1, 2, 3, 5, 8)

```
scala> def f(list1: List[Any]): List[Any] = list1 match {
    case Nil => Nil
    case (x: List[Any]) :: new_el => f(x) ::: f(new_el)
    case x :: new_el => x :: f(new_el)
  }
      |       |       |       | def f(list1: List[Any]): List[Any]

scala> f(List(List(1, 1), 2, List(3, List(5, 8))))
val res5: List[Any] = List(1, 1, 2, 3, 5, 8)
```

20. f(n) - сумма цифр наибольшего простого делителя натурального числа n.

```
scala> def sum_of_num1(n: Int): Int = (1 to n).filter(x=> (n % x == 0 && (! ((2 until x-1) exists (x % _ == 0))))).max.toString.toArray.map(y => y.asDigit)
 .sum
def sum_of_num1(n: Int): Int

scala> sum_of_num1(14)
val res0: Int = 7

scala> sum_of_num1(122)
val res1: Int = 7

scala> sum_of_num1(17)
val res2: Int = 8
```

21. Список содержит элементы одного, но любого типа. Получить список, содержащий каждый имеющийся элемент старого списка k раз подряд. Число k задается при выполнении программы.

```
scala> def copy_list(k: Int, list1: List[Any]): List[Any] = list1.flatMap(List.fill(k)(_))
def copy_list(k: Int, list1: List[Any]): List[Any]

scala> copy_list(3, List("Hello"))
val res4: List[Any] = List(Hello, Hello, Hello)
```

22. Повтор 20

23. Повтор 21

24. f(m,n) - наименьшее общее кратное натуральных чисел m и n.

```
scala> def nod(a: Int, b: Int):Int = if (b == 0) return a else return nod(b,a%b)
def nod(a: Int, b: Int): Int

scala> def nod(a: Int, b: Int):Int = if (b == 0) return a else return nod(b,a%b)
def nod(a: Int, b: Int): Int

scala> def nok(m: Int, n: Int):Int = m*n/nod(m,n)
def nok(m: Int, n: Int): Int

scala> nok(2,3)
val res3: Int = 6

scala> nok(2,5)
val res4: Int = 10
```

25. Список содержит элементы одного, но любого типа. Получить список, из элементов исходного, удаляя каждый k-й элемент. Число k задается при выполнении программы.

```
scala> def del_el(list: List[Any], k: Int):List[Any] = (list.zipWithIndex.filter(x => (1 + x._2)%k != 0).map(_._1))
def del_el(list: List[Any], k: Int): List[Any]

scala> del_el(List('a', 'b','c','d','e'),2)
val res7: List[Any] = List(a, c, e)
```

26. f(n,k) - число размещений из n по k. Факториал не использовать

```
scala> def factor(n: Int, k: Int):Int = (1 to n).foldLeft(1)(_*_)/(1 to (n - k)).foldLeft(1)(_*_)
def factor(n: Int, k: Int): Int

scala> factor(8,3)
val res8: Int = 336
```

27. Список содержит элементы одного, но любого типа. Получить новый список, перемещая циклически каждый элемент на k позиций влево (при перемещении на одну позицию первый элемент становится последним, второй первым и так далее). Число k задается при выполнении программы. Если k отрицательное, то перемещение происходит вправо.

```
scala> def f(list: List[Any], k: Int): List[Any] = list.drop((list.length + k) % list.length) ++ list.take((list.length + k) % list.length)
def f(list: List[Any], k: Int): List[Any]

scala> f(List(1,2,3,4,5,6), 1)
val res9: List[Any] = List(2, 3, 4, 5, 6, 1)

scala> f(List(1,2,3,4,5,6), 5)
val res10: List[Any] = List(6, 1, 2, 3, 4, 5)

scala> f(List(1,2,3,4,5,6), -1)
val res11: List[Any] = List(6, 1, 2, 3, 4, 5)
```

28. f(n) - наибольшее совершенное число не превосходящее n. Совершенным называется натуральное число n равное сумме своих делителей, меньших n, например 6 = 1 + 2 + 3 (f(6) = 6, f(7) = 6, ...).

```
scala> def funk(n: Int):Boolean = (1 to n-1).filter(n % _ == 0).sum == n
def funk(n: Int): Boolean

scala> def f1(n:Int): Int = (1 to n).filter(funk(_)).max
def f1(n: Int): Int

scala> def f1(n:Int): Int = (1 to n).filter(funk(_)).max
def f1(n: Int): Int

scala> f(8)
        ^
       error: not enough arguments for method f: (list: List[Any], k: Int): List[Any].
       Unspecified value parameter k.

scala> f1(8)
val res13: Int = 6

scala> f1(6)
val res14: Int = 6
```

29. Список содержит элементы одного, но любого типа. Получить два списка из элементов исходного, выбирая в первый элементы с четными индексами, а во второй с нечетными.

```
scala> def f_first(list: List[Any]): List[Any] = list.zipWithIndex.filter(x => x._2 % 2 != 0).unzip._1
def f_first(list: List[Any]): List[Any]

scala> def f_second(list: List[Any]): List[Any] = list.zipWithIndex.filter(x => x._2 % 2 == 0).unzip._1
def f_second(list: List[Any]): List[Any]

scala> def f_second(list: List[Any]): List[Any] = list.zipWithIndex.filter(x => x._2 % 2 == 0).unzip._1
def f_second(list: List[Any]): List[Any]

scala> f_first(List(1,2,3,4,5,6,7,8))
val res17: List[Any] = List(2, 4, 6, 8)

scala> f_second(List(1,2,3,4,5,6,7,8))
val res18: List[Any] = List(1, 3, 5, 7)
```

30. f(n) - наибольшее из чисел от 1 до n включительно, обладающее свойством: сумма цифр n в некоторой степени > 1 равна самому числу n. Пример: 512 = 83

```
scala> def f1(n:Int):Int = {
   var sum = 0
   n.toString.foreach(sum += _.asDigit)
   sum
 }
     |      |      |      | def f1(n: Int): Int

scala> import scala.math.pow

 def f2(n:Int):Boolean = {
   var i = 2
   while (pow(f1(n),i)<n){ i += 1}
   pow(f1(n), i) == n
 }
import scala.math.pow

scala>
scala>      |      |      |      | def f2(n: Int): Boolean
```

```
scala> def f(n: Int): Int = {
    var i:Int = n
    while(!f2(i)){
      i-=1
    }
    i
  }
     |      |      |      |      |      | def f(n: Int): Int

scala> f(512)
val res0: Int = 512
```

31. Список в качестве элементов содержит кортежи типа: (n, s), где n — целые числа, а s — строки. Получить два списка из элементов исходного, выбирая в первый числа, а во второй строки из кортежей.

```
scala> List((1, 'A'), (2, 'B'), (3, 'C'), (4, 'D')).unzip._1
val res4: List[Int] = List(1, 2, 3, 4)

scala> List((1, 'A'), (2, 'B'), (3, 'C'), (4, 'D')).unzip._2
val res5: List[Char] = List(A, B, C, D)
```