БГУИР

Кафедра ЭВМ

Отчет по лабораторной работе №1

Тема: «Создание простых S-проектов на основе классов»

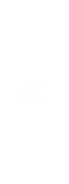
Проверила:

Герман Ю.О.

Выполнили:

Климович А.Н.

Тамашеня В.В.

Минск 2023



**1 ЦЕЛЬ**

Изучить технику использования классов в Scala.

**2 КРАТКИЕ ТЕОРИТИЧЕСКИЕ СВЕДЕНИЯ**

Scala – это статически типизированный язык программирования, который сочетает в себе функциональное и объектно-ориентированное программирование.

Scala является компилируемым языком программирования, что означает, что код на Scala должен быть скомпилирован в байт-код Java перед его выполнением. Scala также имеет интерпретатор, который позволяет быстро тестировать код без необходимости компиляции.

Scala – масштабируемый язык программирования т.к. он был разработан для крупномасштабной разработки ПО. Данный язык способен разбираться с большим объемом данных без увеличения затрат памяти. Также Scala имеет поддержку параллелизма, что позволяет асинхронно обрабатывать несколько потоков. Scala совместима с Java, работает на JVM.

**3 ХОД РАБОТЫ**

**3.1 Анализ примеров**

Пример 1: передать в функцию строку name для последующего вывода строки “Hello, $name!”.

Далее представлен код примера 1:

class Person {

def hello(name: String): Unit = println(s"Hello, $name!")

}

object Main22 {

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

val x = new Person()

x.hello("Alexey")

x.hello("Vladislav")

}

}

Результат выполнения данного примера представлен на рисунке 3.1:

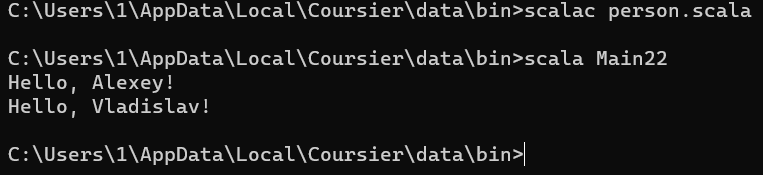


Рисунок 3.1 – Результат выполнения примера 1

Пример 2: ввести с клавиатуры два целых числа и найти их наибольший целый (общий) делитель.

Далее представлен код примера 2:

object Numbers {

def gcd(a: Int, b: Int): Int = {

if (b == 0) a else gcd(b, a % b)

}

}

object Main22 {

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

print("Enter the first number: ")

val a = scala.io.StdIn.readLine().toInt

print("Enter the second number: ")

val b = scala.io.StdIn.readLine().toInt

val gcdValue = Numbers.gcd(a, b)

println(s"The greatest common divisor of $a and $b is $gcdValue")

}

}

Результат выполнения данного примера представлен на рисунке 3.2:

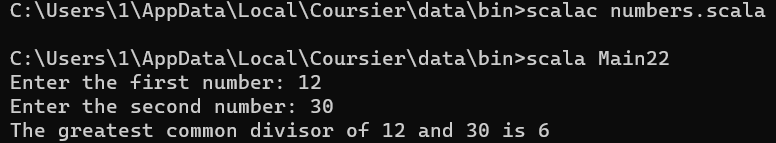


Рисунок 3.2 – Результат выполнения примера 2

Пример 3: определить сумму цифр в записи целого числа.

Далее представлен код примера 3:

object Main22 {

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

print("Enter the number: ")

val a = scala.io.StdIn.readInt

var sum = 0

for (digit <- a.toString) {

sum += digit.asDigit

}

println(sum)

}

}

Результат выполнения данного примера представлен на рисунке 3.3.

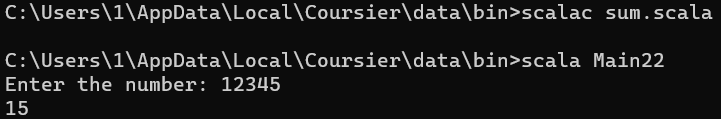


Рисунок 3.3 – Результат выполнения примера 3

Пример 4: сформировать целое число из первых трех цифр шестизначного числа.

Далее представлен код примера 4:

object Main22 {

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

print("Enter the number: ")

val number = scala.io.StdIn.readInt

try{

val numberString = number.toString

val firstThreeDigits = if (numberString.length >= 3 && number >= 0) {

numberString.substring(0, 3).toInt

} else {

throw new Exception("Number does not have at least three digits or is negative")

}

println(firstThreeDigits)

} catch {

case e: Exception => println(e.getMessage)

}

}

}

Результат выполнения данного примера представлен на рисунке 3.4.

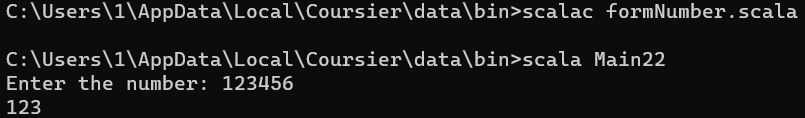


Рисунок 3.4 – Результат выполнения примера 4

**3.2 Вариант 1**

Построить генератор случайных чисел по следующей схеме. Вводите 10-значное число с клавиатуры. Формируете два новых целых числа A и B: первое A состоит из первых пяти цифр, второе B – из последних цифр введенного числа, начиная с шестой. Перемножаете числа A и B друг на друга. Первые три цифры результата С, будучи поделенными на 1000, дают первое случайное число REZ. Чтобы сформировать следующее случайное число, прибавляете к Z число С и повторяете процесс. Формируете два новых целых числа A и B: первое A состоит из первых пяти цифр, второе B – из последних цифр числа Z, начиная с шестой. Перемножаете числа A и B друг на друга. Первые три цифры результата С, будучи поделенными на 1000, дают второе случайное число. Чтобы сформировать следующее случайное число, прибавляете к Z число С и повторяете процесс. Сформируйте 5 случайных чисел.

Если z отрицателен, то это значит, что надо заменить первую цифру единицы на ноль и убрать знак “минус”, кроме того, нужно поддерживать длину строки не меньше 10 символов.

Далее представлен код варианта 1:

object Main {

def main(args: Array[String]): Unit = { // основной метод входа в программу

var numberString = ""

var numberA = 0

var numberB = 0

var numberC = 0

var numberZ = 0

var result: Float = 0.0f

print("Enter the number: ") // приглашение на ввод числа

var flag = 0

while(numberString.length() != 10 || flag == 1){

flag = 0

numberString = scala.io.StdIn.readLine() //считывание «строки» числа

if (numberString.length() != 10) { //если число не 10значное

println("Incorrect number, try again");

}else {

for (i <- 0 to 9) {

if (numberString.charAt(i) < '0' || numberString.charAt(i) > '9') {

//если не все символы строки цифры - повтор ввода

println("Incorrect number, try again");

flag = 1

}

}

}

}

for (i <- 1 to 5) { //цикл для генерации пяти случайных чисел

println("numberString = " + numberString)

numberA = numberString.substring(0, 5).toInt //формирование числа A

numberB = numberString.substring(5, 10).toInt //формирование числа B

println("A = " + numberA)

println("B = " + numberB)

numberC = numberA \* numberB // формирование числа С

println("C = " + numberC)

// формирование случайного числа

result = numberC.toString().substring(0, 3).toFloat / 1000

println("rez = " + result)

numberZ += numberC // для формирования следующего числа

println("z = " + numberZ)

if (numberZ < 0) {

numberZ = ("0" + (numberZ.toString().substring(1, numberZ.toString().

length() - 1))).toInt

}

while(numberZ.toString.length < 10) {

numberZ += 1000000000

}

numberString = numberZ.toString()

println("--------------------------------")

}

}

}

Результат выполнения варианта 1 представлен на рисунке 3.5.

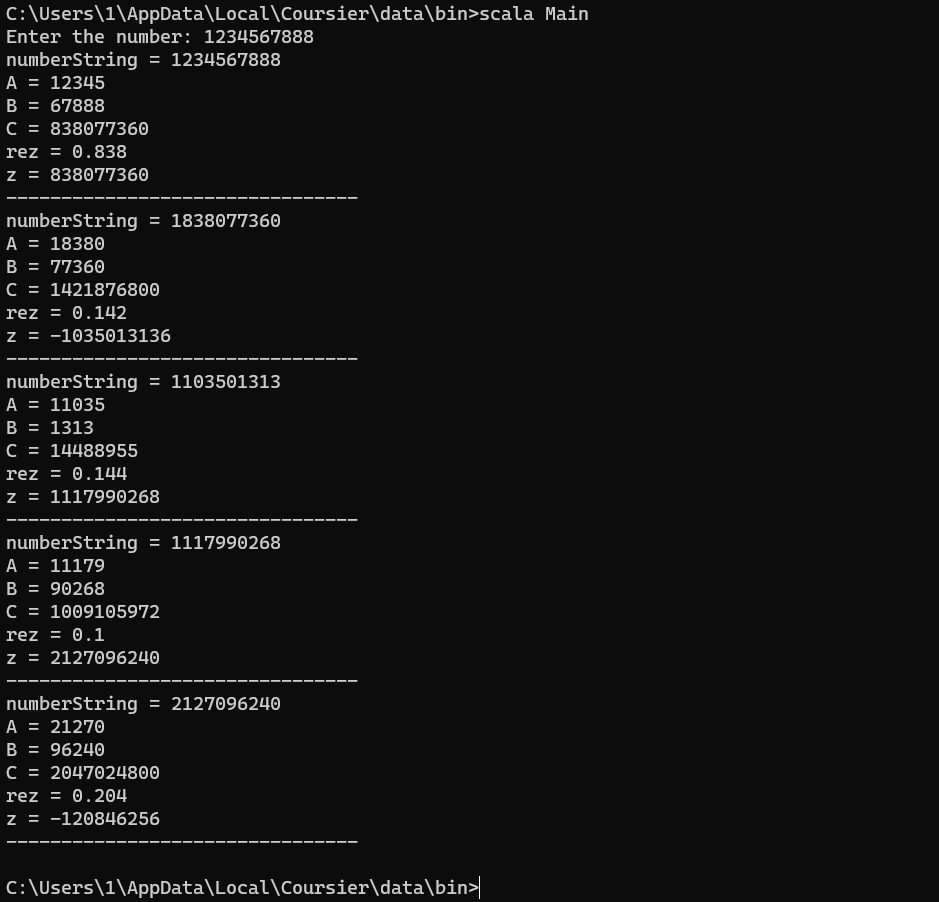


Рисунок 3.5 – Результат выполнения варианта 1

**3.2 Вариант 3**

Построить генератор случайных чисел по следующей схеме. Вводите 10-значное число Z с клавиатуры. Формируете два новых целых числа A и B: первое A состоит из первых пяти цифр, второе B – из последних цифр числа Z, начиная с шестой. Дописываете к числу В число А. Получаете число W. Находите С= Z\*W. Первые три цифры результата С, будучи поделенными на 1000, дают первое случайное число REZ. Чтобы сформировать следующее случайное число, формируете два новых целых числа A и B: первое A состоит из первых пяти цифр, второе B – из последних цифр числа C, начиная с шестой. Дописываете к числу В число А. Получаете число W. Находите C= C\*W. Первые три цифры результата C, будучи поделенными на 1000, дают второе случайное число REZ. Сформируйте 3 случайных чисел по этой схеме.

Далее представлен код варианта 3:

object Main234 {

def func(numberString: String): String = {

var w = 0L

var z = 0L

var c = 0L

var aString = ""

var bString = ""

var cString = ""

var rez: Float = 0.0F

z = numberString.toLong

println("z = " + z)

aString = numberString.substring(0, 5)

bString = numberString.substring(5, 10)

println("a = " + aString)

println("b = " + bString)

w = (bString + aString).toLong

println("w = " + w)

cString = (z \* w).toString()

if(cString.toLong < 0)

cString = cString.substring(1,11)

else

cString = cString.substring(0, 10)

println("c = " + cString)

rez = cString.substring(0, 3).toFloat / 1000

println("rez = " + rez)

println("-----------------------------")

cString

}

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

print("Enter the number: ")

var numberString = ""

var flag = 0

while(numberString.length() != 10 || flag == 1){

flag = 0

numberString = scala.io.StdIn.readLine()

if (numberString.length() != 10) {

println("Incorrect number, try again");

}else {

for (i <- 0 to 9) {

if (numberString.charAt(i) < '0' || numberString.charAt(i) > '9') {

println("Incorrect number, try again");

flag = 1

}

}

}

}

for (i <- 1 to 3) {

numberString = func(numberString)

}

}

}

Результат выполнения варианта 3 представлен на рисунке 3.6.

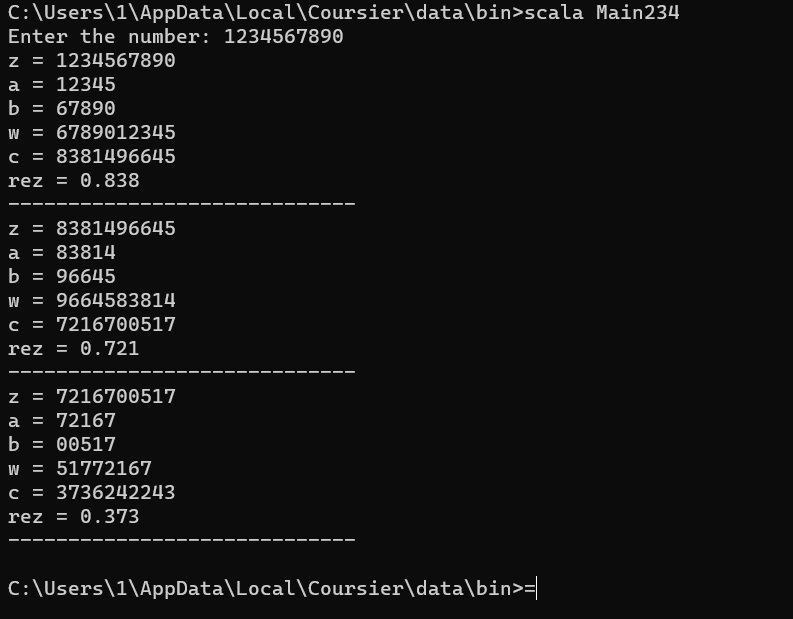


Рисунок 3.6 – Результат выполнения варианта 3

**4 ВЫВОД**

Изучили технику использования классов в Scala.