

Министерство Образования и науки Республики Казахстан
Казахский Национальный Университет имени аль-Фараби
Факультет информационных технологий
Кафедра Информационных систем



СРС

Руководитель курса: Бедельбаев А.А.

Студент ИС 18-10(R&D): Мамбеталина Ә.А.

Алматы 2021

СОДЕРЖАНИЕ

1. Блок А (Тест).....	3-4
2. Блок Б (Открытый вопрос).....	5-7
3. Блок В (Задача).....	8-13
4. Блок Д (Загрузка ответов).....	14

Блок А

(Тест, выберите правильный ответ): (9 балл)

1. Что напечатает следующий фрагмент кода?

```
case class KaznuStudent(name: String, age: Int)
val aidar1=new KaznuStudent("Aidar", 22)
val aidar2=new KaznuStudent("Aidar", 22)
println(bob1==bob2)
```

а) False

б) True

с) Выдаст ошибку (throw error)

д) Выдаст исключение (throw Exception)

2. Следующие утверждения верны для сопутствующих объектов и сопутствующих классов:

а) Сопутствующий объект-это объект с тем же именем, что и класс

б) Сопутствующие классы и объекты могут получить доступ к частным членам своих компаньонов

с) Оба вышеперечисленных

3. Что такое функция высшего порядка в Scala?

а) Он принимает другие функции в качестве параметров

б) В результате он возвращает функцию

с) Оба вышеперечисленных

4. Выделите правильные утверждения из следующих:

а) Классы Case (case class) позволяют сопоставлять паттерны(поиск по шаблону - pattern matching)

б) Мы должны использовать ключевое слово для создания экземпляра класса case(case class)

с) Мы должны вручную определить методы доступа для всех аргументов конструктора

д) Мы должны сгенерировать методы equals(), hashCode () и toString()

5. Что содержит переменная x в следующем коде:

```
var x,y,z=(1,2,3)
```

а) 1

б) (1,2,3)

с) Код выдаст ошибку

6. Коллекция типа collection.Seq неизменная(immutable)

а) False

б) True

Блок Б

(Открытый вопрос) (12 балл)

Вопрос 1. Перечислите разницу между объектом и классом ?

Класс - это определяемая пользователем схема, содержащая поля и методы, которые определяют функциональность использования его полей и методов.

Объект - в объектно-ориентированном программировании объекты используются в реальной жизни. Это экземпляры класса, которые созданы для использования этого класса в программе.

Разница между классами и объектами:

- Класс - это план, а объект - это экземпляр. У классов есть поля и методы, тогда как у объектов есть состояния, поведения и идентичность.
- Объект может существовать соло, т.е. без класса. Чтобы использовать члены класса, нам нужно либо создать объект этого класса, либо расширить его другим классом.
- И класс, и объект могут расширять один класс и одну или две черты. Но абстрактным может быть только класс, а не метод. Одним из основных отличий является наследование, а объекты уникальны, поэтому они не могут быть унаследованы, тогда как классы могут быть унаследованы.
- Определение : класс определяется с помощью ключевого слова `class`, а объект определяется с помощью ключевого слова `object` . Кроме того, в то время как класс может принимать параметры , объект не может принимать никаких параметров.
- Чтобы создать экземпляр регулярного класса, мы используем `new` ключевое слово. Для объекта нам не нужно ключевое слово `new`.
- В то время как класс может иметь неограниченное количество экземпляров, объект имеет только один экземпляр, созданный лениво, когда мы впервые на него ссылаемся.

Вопрос 2. Что такое “Trait” в языке Scala, перечислите особенности?

Trait используются для разделения интерфейсов и полей между классами. Они похожи на интерфейсы Java 8. Но они более мощные, чем интерфейс в Java, потому что в трейтах нам разрешено реализовывать члены. Признаки могут иметь методы (как абстрактные, так и не абстрактные) и поля в качестве своих членов.

Особенности Trait:

- Trait создаются с использованием ключевых слов признаков .
- Trait не содержит параметров конструктора.
- Когда класс наследует один Trait, используем ключевое слово `extends`
- Когда класс наследует несколько признаков, используем ключевое слово `extends` перед первым признаком, а после этого используем ключевое слово `with` перед другими признаками.
- Абстрактный класс также может наследовать Trait с помощью ключевого слова `extends` .
- В Scala один Trait может наследовать другой, используя ключевое слово `extends`
- Trait поддерживают множественное наследование.
- Мы также можем добавить признаки к экземпляру объекта. Или, другими словами, мы можем напрямую добавить Trait в объект класса, не наследуя этот Trait в классе. Мы можем добавить Trait в экземпляр объекта, используя ключевое слово `with` .

Вопрос 3. Что такое Case Class, перечислите особенности?

Еще одна функция Scala, обеспечивающая поддержку функционального программирования, - это класс case . Класс case имеет все функции обычного класса и многое другое. Когда компилятор видит case ключевое слово перед class, он генерирует для нас код со следующими **особенностями**:

- Параметры конструктора класса типа val по умолчанию являются общедоступными полями, поэтому методы доступа создаются для каждого параметра.
- Apply метод создан в сопутствующем объекте класса, так что нам не нужно использовать new ключевое слово , чтобы создать новый экземпляр класса.
- Генерируется unapply метод, который позволяет использовать классы case в match выражениях по-разному .
- В сору классе создается метод. Мы не можем использовать эту функцию в коде Scala / OOP, но в Scala / FP она используется постоянно.
- Equals и hashCode создаются методы, которые позволяют сравнивать объекты и легко использовать их в качестве ключей на картах.
- Создается toString метод по умолчанию , который полезен для отладки.

Самое большое преимущество

Хотя все эти функции являются большими преимуществами для функционального программирования, как написано в книге

«Программирование на Scala (Одерски, Спун и Веннерс)», «самым большим преимуществом классов case является то, **что они поддерживают сопоставление с образцом (pattern matching)**».

Сопоставление с образцом - основная функция языков программирования FP, и классы case Scala предоставляют простой способ реализовать сопоставление с образцом в выражениях сопоставления и других областях.

Блок В

(Задача, требуется приложить ответ компилятора помимо решения самой задачи) (10 балл)

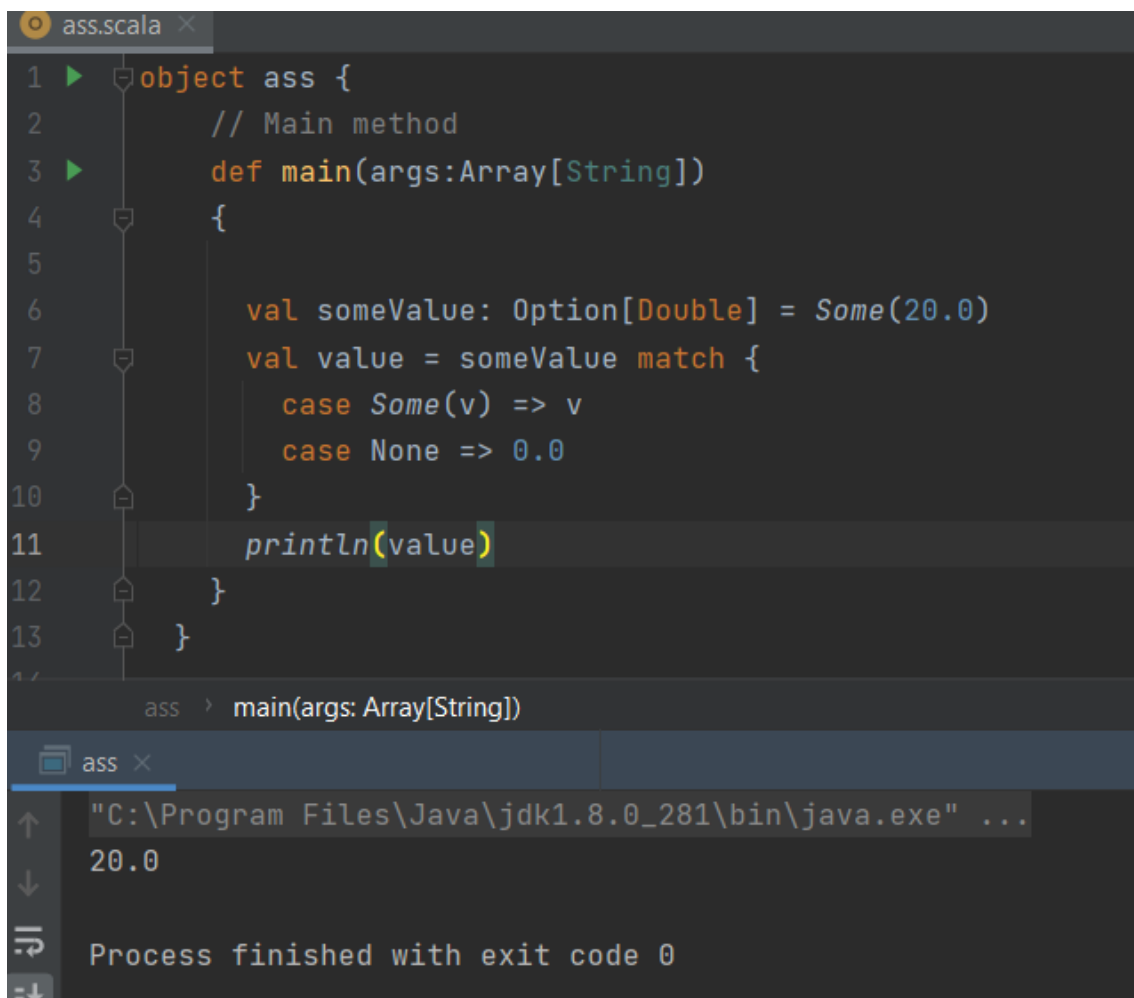
Задача 1.

Дано:

```
val someValue: Option[Double] = Some(20.0)
val value = someValue match {
  case Some(v) => v
  case None => 0.0
}
```

Найти значение “value”, чему равен?

Ответ: 20.0



The screenshot shows an IDE window titled 'ass.scala' with the following Scala code:

```
1  object ass {
2      // Main method
3      def main(args:Array[String])
4      {
5
6          val someValue: Option[Double] = Some(20.0)
7          val value = someValue match {
8              case Some(v) => v
9              case None => 0.0
10         }
11         println(value)
12     }
13 }
```

Below the code editor, the console output is visible:

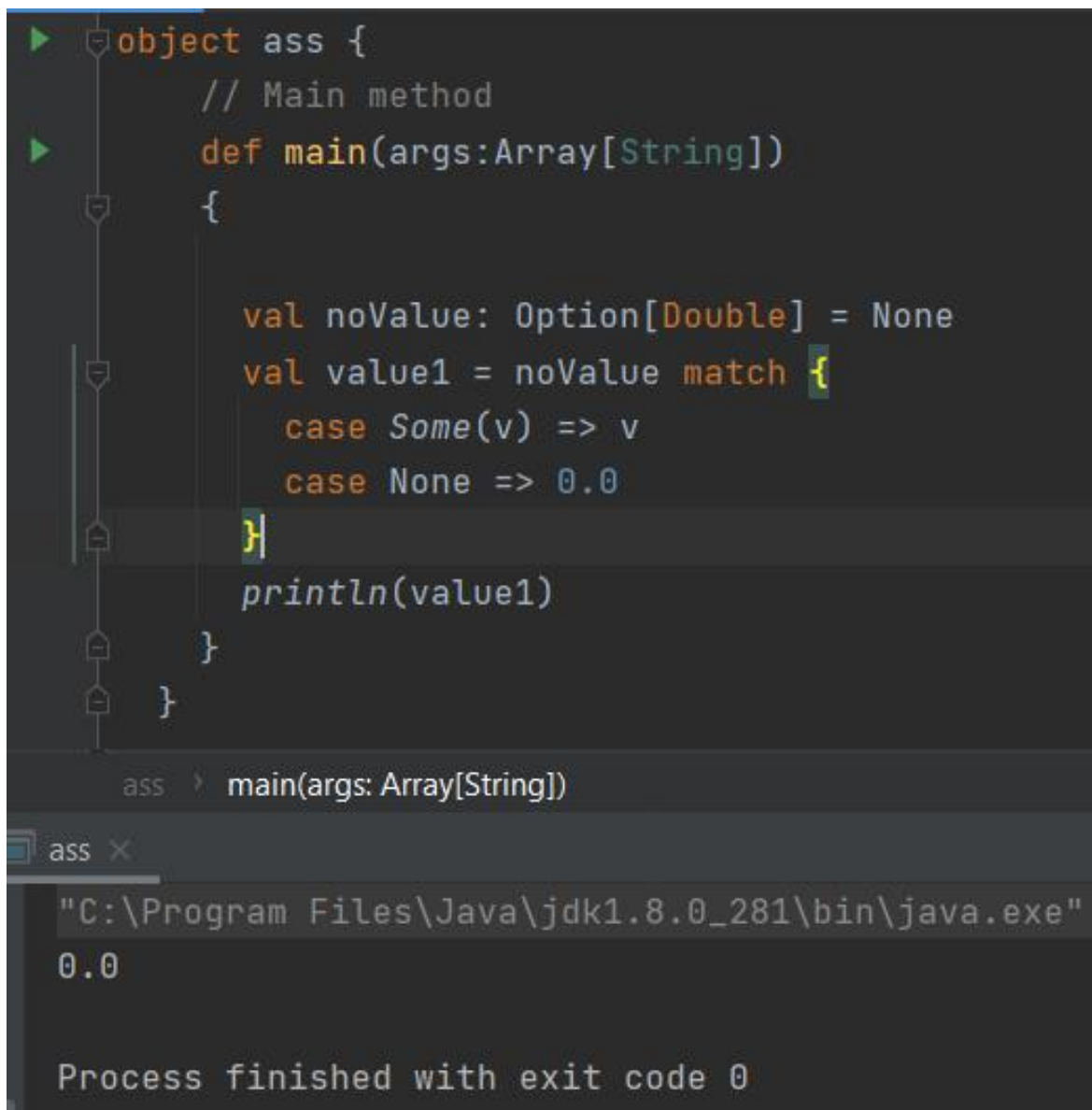
```
ass > main(args: Array[String])
20.0
Process finished with exit code 0
```



```
val noValue: Option[Double] = None
val value1 = noValue match {
case Some(v) => v
case None => 0.0
}
```

Найти значение “value1”, чему равен?

Ответ: 0.0



The screenshot shows an IDE with a Scala file named 'ass'. The code defines an object 'ass' with a 'main' method. Inside 'main', it declares 'noValue' as an 'Option[Double]' set to 'None', then uses a 'match' expression to assign 'value1'. The 'match' has two cases: 'Some(v)' returning 'v' and 'None' returning '0.0'. Since 'noValue' is 'None', 'value1' is assigned '0.0'. The code then prints 'value1'. The IDE's output window shows the command 'C:\Program Files\Java\jdk1.8.0_281\bin\java.exe' and the output '0.0', followed by 'Process finished with exit code 0'.

```
object ass {
  // Main method
  def main(args:Array[String])
  {
    val noValue: Option[Double] = None
    val value1 = noValue match {
      case Some(v) => v
      case None => 0.0
    }
    println(value1)
  }
}
```

ass > main(args: Array[String])

ass x

"C:\Program Files\Java\jdk1.8.0_281\bin\java.exe"

0.0

Process finished with exit code 0

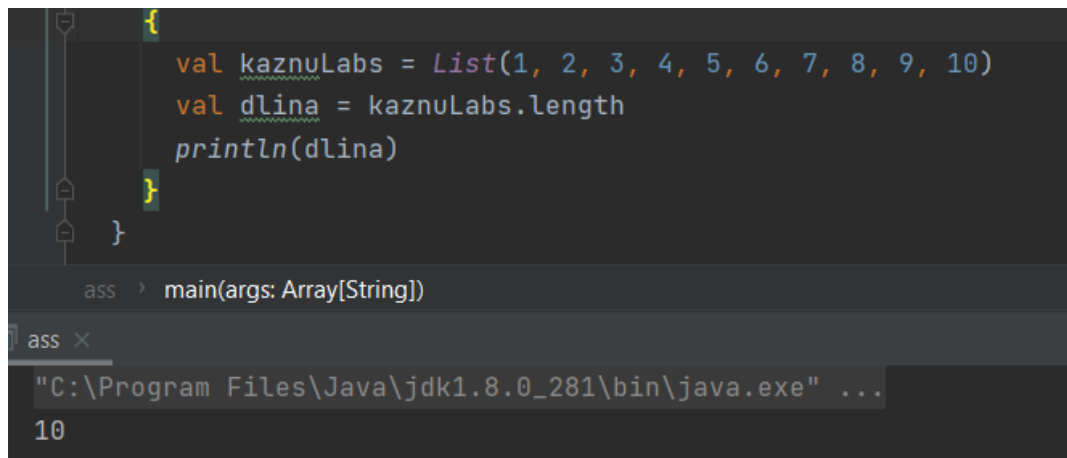
Задача 2.

Задан следующий список: `val kaznuLabs = List(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)`

нужно найти:

- длину списка

Ответ: 10



```
{
  val kaznuLabs = List(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
  val dlina = kaznuLabs.length
  println(dlina)
}
```

ass > main(args: Array[String])

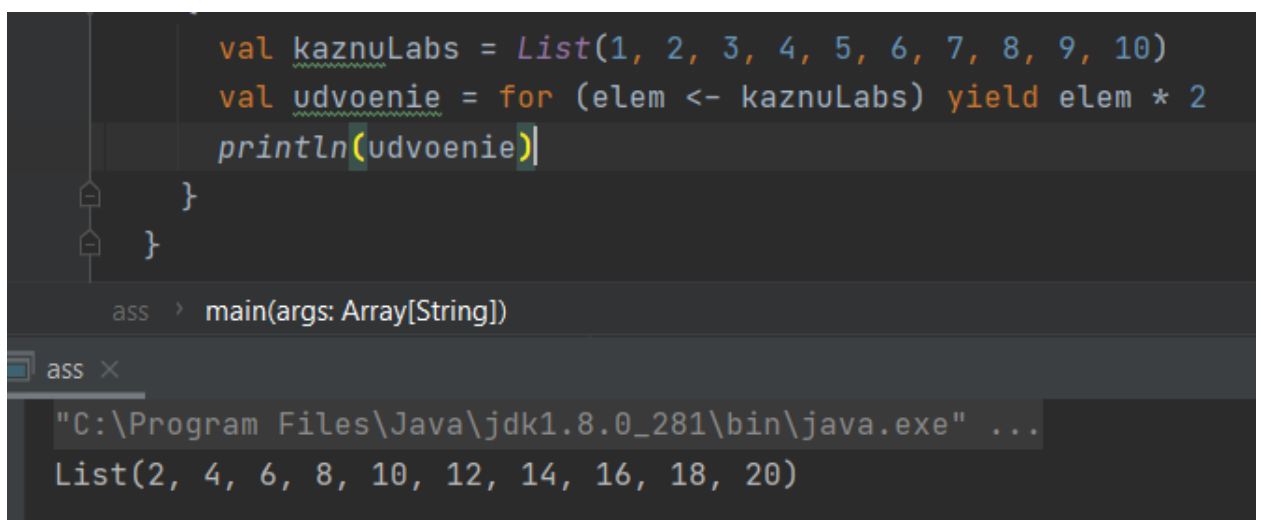
ass x

"C:\Program Files\Java\jdk1.8.0_281\bin\java.exe" ...

10

- удвоить значение элементов списка

Ответ: List(2, 4, 6, 8, 10, 12, 14, 16, 18, 20)



```
val kaznuLabs = List(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
val udvoenie = for (elem <- kaznuLabs) yield elem * 2
println(udvoenie)
```

}

ass > main(args: Array[String])

ass x

"C:\Program Files\Java\jdk1.8.0_281\bin\java.exe" ...

List(2, 4, 6, 8, 10, 12, 14, 16, 18, 20)

- перевернуть список

Ответ: List(10, 9, 8, 7, 6, 5, 4, 3, 2, 1)

```
{
    val kaznuLabs = List(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
    val p = kaznuLabs.reverse
    println(p)
}

ass > main(args: Array[String])
```

ass ×

"C:\Program Files\Java\jdk1.8.0_281\bin\java.exe" ...
List(10, 9, 8, 7, 6, 5, 4, 3, 2, 1)

- отфильтруйте все значения в списке, кратные 3

Ответ: List(3, 6, 9)

```
{
    val kaznuLabs = List(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
    val result = kaznuLabs.filter(_% 3 == 0)
    println(result)
}

ass > main(args: Array[String])
```

ass ×

"C:\Program Files\Java\jdk1.8.0_281\bin\java.exe" ...
List(3, 6, 9)

Задача 3.

Вычислите правильный выход:

```
val numbers=List(11,22,33)
```

```
var total=0
```

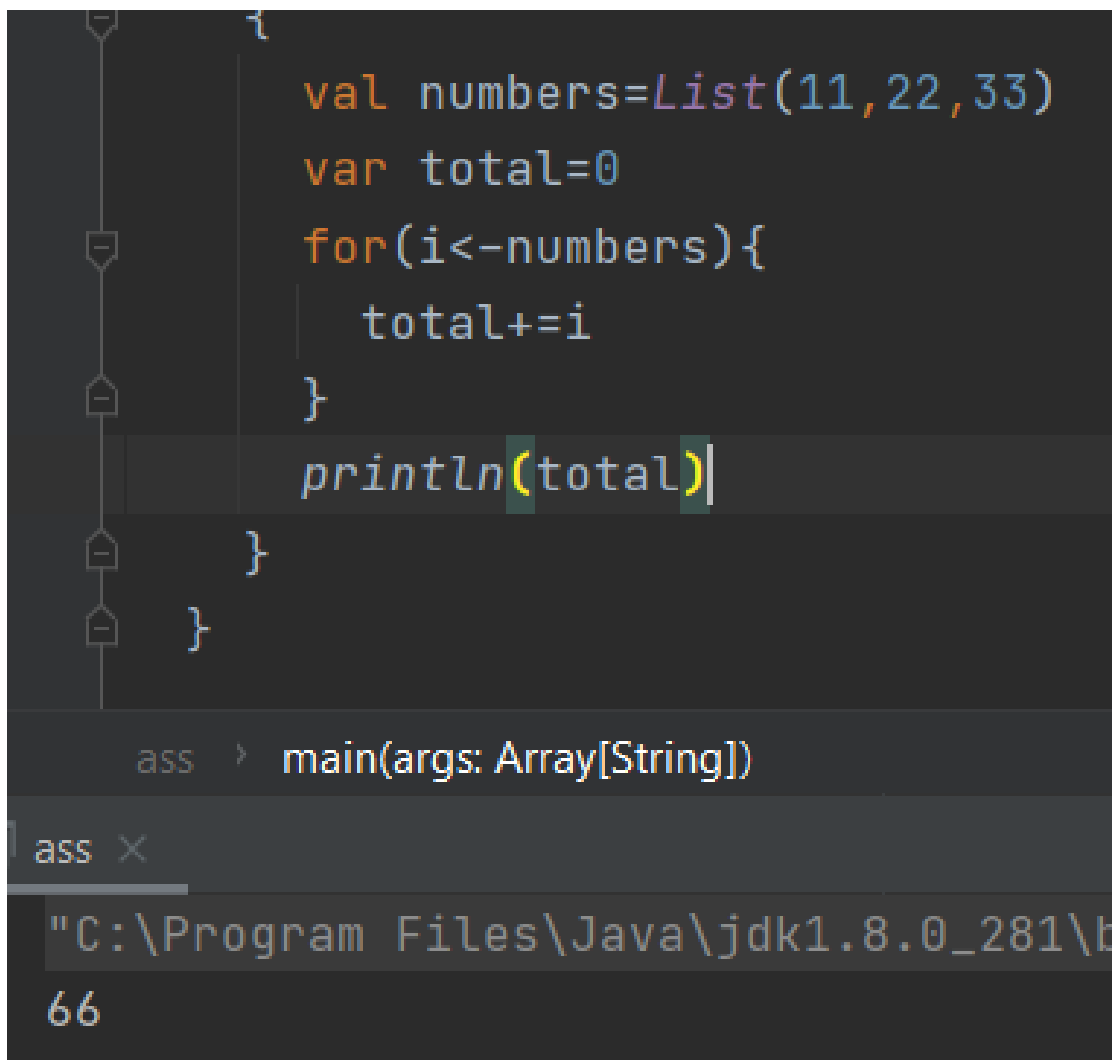
```
for(i<-numbers){
```

```
total+=i
```

```
}
```

```
println(total)
```

Ответ: 66 (Здесь сложение трех цифр(11+22+33))



```
val numbers=List(11,22,33)
var total=0
for(i<-numbers){
  total+=i
}
println(total)
```

ass > main(args: Array[String])

ass x

"C:\Program Files\Java\jdk1.8.0_281\bin\java.exe" -cp "C:\Program Files\Java\jdk1.8.0_281\bin\java.exe" 66

Задача 4.

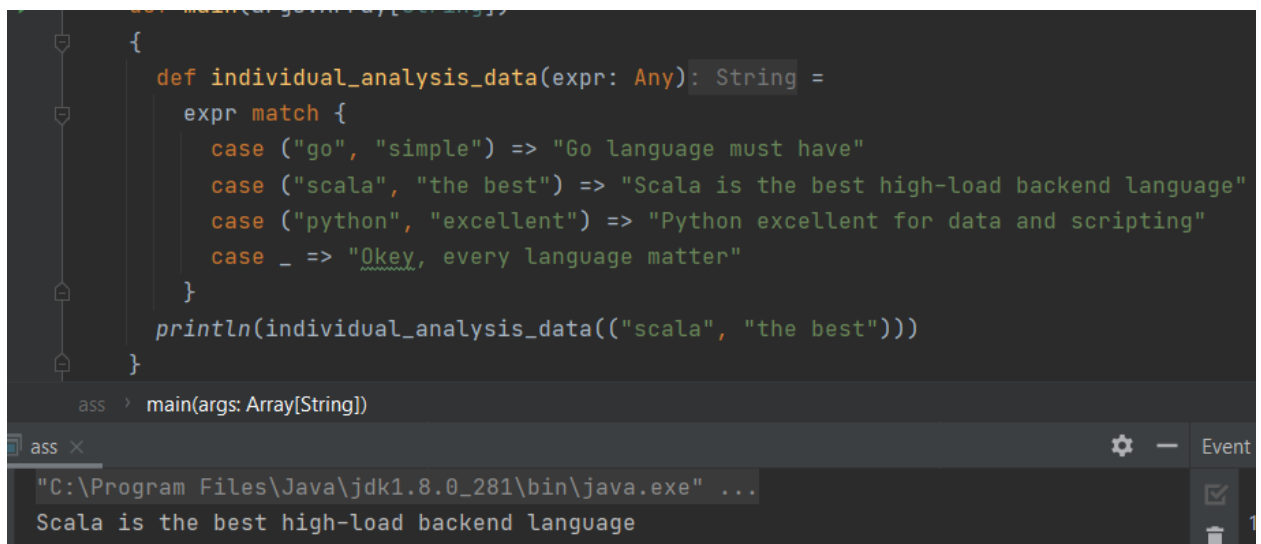
Определите следующий метод, если задан параметр

`individual_analysis_data(("scala", "the best"))` где метод `individual_analysis_data`:

```
def individual_analysis_data(expr: Any) =  
  expr match {  
    case ("go", "simple") => "Go language must have"  
    case ("scala", "the best") => "Scala is the best high-load backend  
    language"  
    case ("python", "excellent") => "Python excellent for data and scripting"  
    case _ => "Okey, every language matter"  
  }
```

Ответ: “Scala is the best high-load backend language”

Если совпадают параметры, выходит соответствующий текст.



The screenshot shows a Scala IDE with a dark theme. The main editor displays the following code:

```
{  
  def individual_analysis_data(expr: Any): String =  
    expr match {  
      case ("go", "simple") => "Go language must have"  
      case ("scala", "the best") => "Scala is the best high-load backend language"  
      case ("python", "excellent") => "Python excellent for data and scripting"  
      case _ => "Okey, every language matter"  
    }  
  println(individual_analysis_data(("scala", "the best")))  
}
```

Below the code, the console output shows the result of the execution:

```
main(args: Array[String])  
Scala is the best high-load backend language
```

Блок Д

(Гитхаб загрузка ответов) (4 балл)

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner *



asemmy ▾



Repository name *

kaznu_asemgul_mambetalina



Great repository names are short and memorable. Need inspiration? How about [friendly-invention?](#)

Description (optional)

CPC SCALA



Public

Anyone on the internet can see this repository. You choose who can commit.



Private

You choose who can see and commit to this repository.