Қазақстан Республикасы білім және ғылым министрлігі

**Әл - Фараби атындағы Қазақ Ұлттық Университеті**



**СӨЖ**

**Пән:** Интеллектуальный анализ данных

**Орындаған:** Сәкенова Әсемгүл

**Алматы 2021 ж**

**Блок А. (Тест, выберите правильный ответ): (9 балл):**

1.с 2.с 3.с 4.а 5.б 6.б

**Блок Б. (Открытый вопрос) (12 балл)**

**Вопрос 1. Перечислите разницу между объектом и классом ?**

Объект - бұл мән немесе бір данасы бар класс деп айтсақ болады. Объектінің анықтамасы классқа ұқсайды, бірақ **object** кілт сөзін қолданады. Жоғары деңгей мәні ретінде объект singleton болып табылады. Оның ең қарапайым жазылу түрі келесідей:

**object** **Box**

Әдіспен объектінің мысалы:

**package** logging **object** **Logger** { **def**  **info**(message: **String**): **Unit** = println(s"INFO: $message") }

Әдіс туралы ақпаратты бағдарламаның кез келген жерінен импорттауға болады.

Серіктес объектілер

Класспен аттас объектіні **серіктес объект** деп атайды. Серіктес класс немесе объект өзінің серігінің жеке компаньонымен қатынаса алады. Серіктес класының даналарына тән емес әдістер мен мәндер үшін серіктес объектіні пайдаланамыз.

Скаладағы класстар - бұл объектілерді құруға арналған схемалар. Оның құрамында әдістер, мәндер, айнымалылар, типтер, объектілер, сипаттамалар және кластар болуы мүмкін.

Ең қарапайым класс анықтамасы - бұл тек class кілт сөзі және идентификатор. Сынып атаулары бас әріптен басталуы керек. Ал **new** кілт сөз - класстың данасын құру үшін қолданылады.

**class** **Person**

**val** user1 = **new** **Person**

Бұл жердегі Person класының стандартты конструкторы бар, ол ешқандай аргументтер қабылдамайды, өйткені конструктор анықталмаған. Дегенмен, бізге көбінесе конструктор мен сынып корпусы қажет. Төменде Point үшін сыныпты анықтаудың мысалы келтірілген:

**class** **Point**(x: **Int**, y: **Int**)

**val** point = **new** **Point**(1, 2)

Point.x

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

Trait- бұл өрістерді немесе басқа кластарға кірістіру үшін қолданылатын әдістерді жинақтайтын объект. Бұл механизм бізге Scala-да бірнеше мұрагерліктің жетіспеушілігін шешуге көмектеседі, біз сыныпта кез-келген трейттерді пайдалана аламыз. Оның ең қарапайым жазылуы **trait** кілт сөзі және оның атауынан тұрады:

**trait** **HairColor**

Trait пайдалану үшін, біз одан класты extends кілт сөзі арқылы мұрагерлікке алуымыз керек. Одан кейін бізге трейттің барлық дерексіз мүшелерін қайта анықтау үшін override кілт сөзін қолданамыз. Оны келесі мысалдан көрсек болады:

**trait** **Iterator**[**A**] {

**def** **hasNext**: **Boolean**

**def** **next**(): **A**

}

**class** **IntIterator**(to: **Int**) **extends** **Iterator**[**Int**] {

**private** **var** current = 0

**override** **def** **hasNext**: **Boolean** = current < to

**override** **def** **next**(): **Int** = {

**if** (hasNext) {

**val** t = current

current += 1

t

} **else** 0

}

}

**val** iterator = **new** **IntIterator**(10)

iterator.next() *// вернет 0*

iterator.next() *// вернет 1*

**Вопрос 3. Что такое Case Class, перечислите особенности?**

Case Class жазылу синтаксисінің ең қарапайым нұсқасы: case class кілт сөзінің көрсетілімі, аты және параметрлер тізімі (олар бос болуы мүмкін). Мысалы:

**case** **class** **Book**(isbn: **String**)

**val** frankenstein = **Book**("978-0486282114")

Кейс-класстар - бұл жай кәдімгі сыныптар, олар:

* әдепкі бойынша өзгермейтін
* Шаблондар арқылы ыдырайтын
* сілтеме орнына құрылымдық теңдікке қарсы
* Түсіндіруге және жұмыс істеуге арналған қысқаша

Case класында объектіні құрумен айналысатын apply() әдісі бар. Параметрлері бар Case классты құру кезінде бұл параметрлер жалпыға қол жетімді және өзгермейді, мысалы:

**сase** **class** **Message**(sender: **String**, recipient: **String**, body: **String**)

**val** message =**Message**("guillaume@quebec.ca", "jorge@catalonia.es", "Ça va ?")

println(message1.sender) *// prints guillaume@quebec.ca*

message1.sender = "travis@washington.us" *// эта строка не компилируется*

**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?

**value=20.0**

val noValue: Option[Double] = None

val value1 = noValue match {

case Some(v) => v

case None => 0.0

}

Найти значение “value1”, чему равен?

**value1=0.0**

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

нужно найти:

- длину списка

**kaznuLabs.length**

**val res1: Int = 10**

- удвоить значение элементов списка

**for(i<-kaznuLabs)print(s"${i \* 2} ")**

**2 4 6 8 10 12 14 16 18 20**

- перевернуть список

**kaznuLabs.reverse**

**val res2: List[Int] = List(10, 9, 8, 7, 6, 5, 4, 3, 2, 1)**

- отфильтруйте все значения в списке, кратные 3

**kaznuLabs.filter(\_%3==0)**

**val res3: List[Int] = 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)

**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"

}

**Scala is the best high-load backend language**

**Блок Д. (Гитхаб загрузка ответов) (4 балл)**

Создайте в гите репозитории(проект) с названием kaznu\_name\_surname, где name и surname ваше имя и фамилия; Залейте вашу работу в созданную гит репозиторию(инструкция доступна на самом сайте github); Важно создавать проект под своим аккаунтом; Важно создать открытый репозитории(public, not private) чтобы мог получить доступ;