МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ВГУ»)

УТВЕРЖДАЮ

Заведующий кафедрой
Кибербезопасности информационных систем
Кенин С. Л.
22.03.2024 г.

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ
Б1.В.05 Разработка мобильных приложений

1. Код и наименование направления подготовки:

09.04.03 Прикладная информатика

2. Профиль подготовки: Прикладная информатика в социальных и медицинских системах

3. Квалификация выпускника: магистр

4. Форма обучения: очная

5. Кафедра, отвечающая за реализацию дисциплины: ПОиАИС

6. Составители программы: Болотова Светлана Юрьевна,
кандидат физико-математических наук, доцент
7. Рекомендована: НМС факультета ПММ, протокол № 5 от 22.03.2024

8. Учебный год:_2024/2025 Семестр: 2

9.Цели и задачи учебной дисциплины
Целями освоения дисциплины являются: изучение основ и получение практических навыков программной инженерии, а также – руководства процессом разработки в области разработки программного обеспечения для мобильных устройств.
Основные задачи преподавания дисциплины следующие:
ознакомление с мобильной операционной системой iOS;
ознакомление с различными инструментами разработки программного обеспечения для мобильных устройств;
знакомство с особенностями разработки мобильных приложений;
изучение основных приемов и методов программирования мобильных приложений, их сравнение и анализ;
знакомство с основными конструкциями соответствующего языка программирования;
получение практических навыков разработки мобильных приложений с применением различных подходов, обеспечивающих оптимизацию и реинжиниринг пользовательских приложений.

10. Место учебной дисциплины в структуре ООП: Дисциплина относится к части, формируемой участниками образовательных отношений, Блока 1. Дисциплины (модули).

11. Планируемые результаты обучения по дисциплине/модулю (знания, умения, навыки), соотнесенные с планируемыми результатами освоения образовательной программы (компетенциями) и индикаторами их достижения:

Код	Название компетенции	Код(ы)	Индикатор(ы)	Планируемые результаты обучения
ПК-2	Непосредственное руководство процессами разработки программного обеспечения	ПК-2.1.	Использует современные инструментальные средства для разработки мобильных приложений	Знать: современные инструментальные средства для разработки мобильных приложений Уметь: использовать современные инструментальные средства для разработки мобильных приложений Владеть: навыками разработки программного обеспечения для мобильных приложений
ПК-3	Управление аналитическими работами	ПК-3.3	Анализирует и сравнивает различные подходы, обеспечивающие оптимизацию и реинжиниринг пользовательских приложений	Знать: различные подходы, обеспечивающие оптимизацию и реинжиниринг пользовательских приложений Уметь: анализировать и сравнивать различные подходы, обеспечивающие оптимизацию и реинжиниринг пользовательских приложений Владеть: навыками управления аналитическими работами, необходимыми при создании мобильных приложений

12. Объем дисциплины в зачетных единицах/час. — 3/108.

Форма промежуточной аттестации зачет с оценкой

13. Трудоемкость по видам учебной работы

Вид учебной работы		Трудоемкость			
		Всего	По семестрам		
			1		…
Аудиторные занятия		50	50		
в том числе:	лекции	16	16		
	практические				
	лабораторные	34	16		
Самостоятельная работа		58	58		
Итого:		108	108		

13.1. Содержание дисциплины

п/п	Наименование раздела	Содержание раздела дисциплины	Реализация

	дисциплины		раздела дисциплины с помощью онлайн-курса, ЭУМК *
1. Лекции			
1.1	Введение в разработку мобильных приложений	Архитектура платформы iOS, компоненты платформы. Знакомство с XCode.	https://edu.vsu.ru/course/view.php?id=18618
1.2	Шаблон проектирования MVC	Шаблон проектирования MVC. Классы, структуры.	https://edu.vsu.ru/course/view.php?id=18618
1.3	Autolayout. Контроль доступа. Перечисления	CountableRange чисел с плавающей точкой, кортежи, вычисляемые свойства, управление доступом, assertions, enum .	https://edu.vsu.ru/course/view.php?id=18618
1.4	Протоколы. Oprionals. Расширения	Протоколы. Oprionals. Расширения.	https://edu.vsu.ru/course/view.php?id=18618
1.5	Строки. Функции. Замыкания	Строки с атрибутами. Функции как типы. Замыкания. Обработка ошибок в Swift. Использование Any в Swift. «Кастинг типа» с помощью оператора as? NSObject, NSNumber, Date, Data.	https://edu.vsu.ru/course/view.php?id=18618
1.6	Views	Views. Создание пользовательского subclass UIView.	https://edu.vsu.ru/course/view.php?id=18618
3. Лабораторные занятия			
3.1	Шаблон проектирования MVC	Демонстрационный пример. Создание проекта Project в Xcode 9 Построение пользовательского интерфейса (UI) iOS симуляторы print (вывод на консоль, используя \( ) нотацию) Определение класса в Swift, включая определение переменных экземпляра класса и методов Связывание свойств (переменных экземпляра класса) в Swift коде с элементами пользовательского интерфейса UI (Outlets) Привязка элементов UI к методам в коде Swift (Actions) Доступ к iOS документации из кода Автоматическое выполнение кода при каждом изменении значения свойства Optionals (?, неявное развертывание путем декларирования со знаком !, явное развертывание с помощью ! и if let) Array Создание модели игры Concentration.	https://edu.vsu.ru/course/view.php?id=18618
3.2	Рисование в iOS	Рисование с помощью Core Graphics и UIBezierPath. Режим contentMode у UIView. Рисование с прозрачностью. Больше ключей для словаря строки с атрибутами NSAttributedString … UIFont и NSParagraphStyle UIFontMetrics масштабирование шрифтов определенного стиля согласное с пользовательскими настройками (Setting) в Larger Text Управление subviews в вашем	https://edu.vsu.ru/course/view.php?id=18618

		пользовательском UIView Использование isHidden Аффинные преобразования CGAffineTransform UIView Приоритеты ограничений (Constraint Priority) в системе Autolayout Assets.xcassets и рисование с помощью UIImage @IBDesignable и @IBInspectable Отображение изображений UIImage в Interface Builder	

13.2. Темы (разделы) дисциплины и виды занятий

№ п/п	Наименование темы (раздела) дисциплины	Виды занятий (количество часов)				
		Лекции	Практические	Лабораторные	Самостоятельная работа	Всего
1	Введение в разработку мобильных приложений	2			2	4
2	Шаблон проектирования MVC.	2		4	2	8
3	Autolayout. Контроль доступа. Перечисления.	2		4	8	14
4	Протоколы. Oprionals. Расширения	2		4	10	14
5	Строки. Функции. Замыкания.	2		6	10	18
6	Views	2		8	12	22
9	Рисование в iOS	4		8	14	26
	Итого:	16		34	58	108

14. Методические указания для обучающихся по освоению дисциплины

рекомендации обучающимся по освоению дисциплины: работа с конспектами лекций, презентационным материалом

15. Перечень основной и дополнительной литературы, ресурсов интернет, необходимых для освоения дисциплины *(список литературы оформляется в соответствии с требованиями ГОСТ и используется общая сквозная нумерация для всех видов источников)*

а) основная литература:

№ п/п	Источник
1	Соколова, В. В. Разработка мобильных приложений : учебное пособие / В. В. Соколова. — Томск : ТПУ, 2014. — 176 с. — ISBN 978-5-4387-0369-3. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/82830 (дата обращения: 27.03.2021). — Режим доступа: для авториз. пользователей.
2	Грэхем, Л. Разработка через тестирование для iOS / Л. Грэхем ; перевод с английского А. Н. Киселев. — Москва : ДМК Пресс, 2013. — 272 с. — ISBN 978-5-94074-863-2. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/63183 (дата обращения: 27.03.2021). — Режим доступа: для авториз. пользователей.

б) дополнительная литература:

№ п/п	Источник
1	Райфельд, М. А. Системы и сети мобильной связи : учебное пособие / М. А. Райфельд, А. А. Спектор. — Новосибирск : НГТУ, 2019. — 96 с. — ISBN 978-5-7782-3833-6. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/152245 (дата обращения: 27.03.2021). — Режим доступа: для авториз. пользователей.
2	Программируем для iPhone и iPad / Д. Пайлон, Т. Пайлон. - Питер, 2012. ISBN 978-5- 459-00375-8.
3	Разработка и продажа программ для iPhone и iPad / Дмитрий Елисеев. - БХВ- Петербург, 2012. ISBN 978-5-9775-0687-8.

в) информационные электронно-образовательные ресурсы (официальные ресурсы интернет)***:**

№ п/п	Ресурс
1	https://edu.vsu.ru/course/view.php?id=18618
2	www.staford.edu Курс лекций Ipad and Iphone Application Development
3	Open handset alliance, http://www.openhandsetalliance.com/ .

* Вначале указываются ЭБС, с которыми имеются договора у ВГУ, затем открытые электронно-образовательные ресурсы, онлайн-курсы, ЭУМК

16. Перечень учебно-методического обеспечения для самостоятельной работы

(учебно-методические рекомендации, пособия, задачники, методические указания по выполнению практических (контрольных), курсовых работ и др.)

№ п/п	Источник
1	Mayer R. Professional Android 4 Application Development. – Willeyand Sons, 2012.
2	*Соколова В. В. Разработка мобильных приложений : учебное пособие. — Томск: Изд-во ТПУ, 2014. — 175 с.: ил.*
3	Таненбаум, Э. Современные операционные системы [Текст] = Modern Operating Systems / Э. Таненбаум ; пер. с англ. Н. Вильчинского, А. Лашкевич. - 3-е изд. - Санкт-Петербург : Питер, 2012. - 1115 с.

17. Образовательные технологии, используемые при реализации учебной дисциплины, включая дистанционные образовательные технологии (ДОТ), электронное обучение (ЭО), смешанное обучение):

При реализации дисциплины используются модульно-рейтинговая и личностно-ориентированные технологии обучения (ориентированные на индивидуальность студента, компьютерные и коммуникационные технологии). В рамках дисциплины предусмотрены следующие виды лекций: информационная, лекция-визуализация, лекция с применением обратной связи.

Дисциплина реализуется с применением электронного обучения и дистанционных образовательных технологий, для организации самостоятельной работы обучающихся используется онлайн-курс, размещенный на платформе Электронного университета ВГУ (LMS moodle), а также другие Интернет-ресурсы, приведенные в п.15в.

18. Материально-техническое обеспечение дисциплины:

Учебная аудитория для проведения лекций, практических занятий, организации самостоятельной работы, проведения текущих и промежуточных аттестаций: специализированная мебель, доска маркерная или меловая, компьютер (ноутбук), мультимедийное оборудование (проектор, экран, средства звуковоспроизведения), допускается использование переносного оборудования.
Для самостоятельной работы необходимы компьютерные классы, помещения, оснащенные компьютерами с доступом к сети Интернет.
Программное обеспечение: Xcode
Материально-техническое обеспечение:
Моноблок Apple iMac MD093RU/A (14 шт.): процессор Intel Core i5 (2.70 GHz), оперативная память 8 Гб, HDD 1 Тб, видеокарта GeForce GT640M 512Мб, диагональ экрана 21,5"
Компьютер APPLE Mac Pro MD772RU/A Xeon W3565 в составе:
системный блок APPLE: процессор Intel Xeon W3565, оперативная память 8Гб, HDD 2Тб, видеокарта AMD Radeon HD 5770
Коммутатор HP ProCurve Switch 1400-24G
Мультимедиа-проектор BENQ MH535
Доска магнитно-маркерная на стенде (100x150см), 2-сторонняя, BRAUBERG PREMIUM

19. Оценочные средства для проведения текущей и промежуточной аттестаций

Порядок оценки освоения обучающимися учебного материала определяется содержанием следующих разделов дисциплины:

№ п/п	Наименование раздела дисциплины (модуля)	Компетенция(и)	Индикатор(ы) достижения компетенции	Оценочные средства
1.	Введение в разработку	ПК-2	ПК-2.1	Лабораторная работа

№ п/п	Наименование раздела дисциплины (модуля)	Компетенция(и)	Индикатор(ы) достижения компетенции	Оценочные средства
	мобильных приложений			
2.	Шаблон проектирования MVC.	ПК-2	ПК-2.1	Лабораторная работа
3.	Autolayout. Контроль доступа. Перечисления.	ПК-3	ПК-3.3	Лабораторная работа
4.	Протоколы. Oprionals. Расширения	ПК-2 ПК-3	ПК-2.1 ПК-3.3	Лабораторная работа
5.	Строки. Функции. Замыкания.	ПК-3	ПК-3.3	Лабораторная работа
6.	Views	ПК-2	ПК-2.1	Лабораторная работа
9.	Рисование в iOS	ПК-2 ПК-3	ПК-2.1 ПК-3.3	Лабораторная работа
Промежуточная аттестация форма контроля – зачет с оценкой				Контрольное тестирование

20 Типовые оценочные средства и методические материалы, определяющие процедуры оценивания

20.1 Текущий контроль успеваемости

Контроль успеваемости по дисциплине осуществляется с помощью следующих оценочных средств: *лабораторная работа.*

Перечень *лабораторных работ*

Реализуйте игру Концентрация, демонстрируемую на лекции. Печатайте весь код, не пользуйтесь копированием и вставкой кода откуда-то. Добавьте больше карт в вашу игру. Добавьте на ваш UI кнопку “New Game”, которая заканчивает текущую игру и начинает новую. В данный момент карты в Модели не рандомизированы (именно поэтому в вашем UI парные карты всегда лежат на тех же самых местах). Перетасуйте карты в методе init() класса Concentration. Введите в игру концепцию “Тема” (“theme”). Тема theme определяет множество эмоджи, из которого выбираются эмоджи для карт. Все эмоджи в определенной теме theme должны иметь отношение к этой теме. Ваша игра должна, по крайней мере, иметь 6 различных тем, и темы должны выбираться случайно каждый раз при старте новой игры. Ваша архитектура должна давать возможность добавлять новую тему одной строкой кода. Добавьте на ваш UI метку для счета в игре (score label). Счет в игре формируется добавлением 2-х очков за каждое совпадение и штрафом в 1 очко за каждое несовпадение ранее увиденной карты. Отслеживание числа переворотов карт flipCount определенно НЕ принадлежит вашему Controller в правильной MVC архитектуре. Исправьте это. Весь новый добавленный UI должен правильно располагаться и выглядеть хорошо в портретном режиме на iPhone X.

Реализуйте игру Set в версии соло (для одного игрока). Разместите на экране по крайней мере 24 карты игры Set. В Set все карты всегда лежат “лицом” вверх. При старте сдайте только 12 карт. Они могут появиться где угодно на экране (то естьнеобязательно их выравнивать по верху или по низу экрана или как-то еще; при старте они могут быть рассеяны, если хотите), но они не должны перекрываться. Вам необходимо также иметь кнопку “Deal 3 More Cards” (Сдай еще 3 карты) (согласно правилам игры Set). Разрешите пользователю выбирать карты касанием для того, чтобы попытаться составить Set. На ваше усмотрение, как показывать “выбор” в вашем UI. Некоторые идеи того, как это можно сделать представлены ниже в подсказках. Также обеспечьте возможность переход из состояние “выбрано” (selected) в состояние “не выбрано” (deselected) (но только когда 1 или 2 (не 3) карты выбраны в данный момент). После того, как выбраны 3 карты, вы должны дать пользователю индикацию, совпали ли эти 3 карты или нет (согласно правилам игры Set). Вы можете сделать это с помощью цвета или как хотите, но пользователю должно быть понятно, совпали эти 3 карты или нет... Когда выбрана новая карта и есть уже 3 выбранных (selected) и не совпавших Set карты, сделайте эти 3 не совпавших карты не выбранными (deselected), а новую карту выбранной (selected). Согласно правилам игры Set, когда выбрана новая карта и есть уже 3 совпавших (matching) и выбранных (selected) Set карты, замените эти 3 совпавших (matching) Set карты новыми из колоды в 81 Set карту (опять, смотрите правила игры Set и что собой представляет колода Set карт). Если

колода пуста, то совпавшие (matching) Set карты не могут быть заменены, но они могут быть скрыты (hidden) в вашем UI. Если вновь выбранная карта является одной из 3-х совпавших (matching) Set карт, то никакие карты не должны быть выбранными (selected) (так как вновь выбранная карта либо будет заменена, либо будет больше невидима на UI). Когда кнопка “Deal 3 More Cards” (Сдай еще 3 карты) нажата, то либо a) происходит замена выбранных карт, если они совпали, либо b) добавляются 3 карты в игру. Кнопка “Deal 3 More Cards” (Сдай еще 3 карты) должна быть недоступна, если a) больше нет карт в Set колоде или b) больше нет мести на UI, чтобы принять еще 3 карты (заметьте, что всегда есть место для размещения еще 3-х карт, если выбранные в данный момент карты совпали (match), так как они заменяются). Вместо рисования Set карт в классической форме, мы будем использовать эти 3 символ ▲ ● ■ и использовать атрибуты в NSAttributedString для соответствующего их рисования (то есть цвета и затенение (shading)). И таким образом, ваши карты могут быть просто кнопками UIButtons. Используйте метод, который берет в качестве аргумента замыкание как значимую часть вашего решения. Используйте перечисление enum как значимую часть вашего решения. Добавьте осмысленное расширение extension к некоторым структурам данных как значимую часть вашего решения. Ваш UI должен иметь прекрасно расположенные UI элементы и хорошо выглядеть (по крайней мере в портретном режиме, желательно также и в ландшафтном режиме, хотя это не обязательно) на любом iPhone 7 или старше. Это означает, что вам следует использовать несколько простых приемов работы с Autolayout, включая Stack Views.

Описание технологии проведения

Каждая лабораторная работа выполняется на основе задания и соответствующей лекции. После выполнения задания на лабораторную работу каждый студент должен выполнить те же действия, но уже по своей теме, которая относится к домашнему заданию по дисциплине. Таким образом, после каждой лабораторной работы формируются необходимые части/знания для выполнения домашнего задания.

Требования к выполнению заданий (или шкалы и критерии оценивания)

Каждая лабораторная работа оценивается по принципу «зачет/незачет»

«Зачет» ставится, если сделано верно не менее 80% задания

«Незачет» ставится, если сделано верно менее 80% задания

20.2 Промежуточная аттестация

Промежуточная аттестация по дисциплине осуществляется с помощью следующих оценочных средств: *контрольное тестирование*

Примеры теста

1. Объясните идею паттерна MVC на примере простейшего калькулятора.
2. Приведите пример создания кортежа с 2 элементами (строкой и целым числом) со значениями “hello” и 12.
3. Как называется такое свойство?

```
var prop: Int {
      get { … }
      set (newValue) {…}
}
```

4. Какой тип имеет переменная a?

```
let a: Int?
```

5. Напишите код, который распечатает значение этой переменной.
6. Исправьте ошибку в данном коде.

```
enum Item {
case A (number: Int)
case B (part: Double)
case C (description: String)
case D

func switchToBeingCookie() {
self = .D
}
}
```

7. В чем заключается ошибка в данном коде?

```
protocol A {
      mutation func f1()
}
```

```
class B: A {
      func f1()
func  f2()
}

let x: B = B()
var y: A = x
y.f2()
```

8. Что означает lazy в данном коде?

```
lazy var a = A()
```

8. Что такое циклическая ссылка в памяти? Использование какого типа ссылок (string, weak, unowned) позволяет цикл разорвать?
9. Что происходит в этом коде?

```
if let cvc = vc as? ConcentrationViewController {…}
```

10. На экране находится прямоугольник, который вы повернули. С помощью какого/каких методов можно заставить прямоугольник перерисоваться:

```
func draw(_ rect: CGRect);
func setNeedsDisplay( _ rect: CGRect);
func redraw(_ rect: CGRect);
func update();
func setNeedsDisplay( )?
```

Описание технологии проведения

Тестирование проходит в письменной форме

Требования к выполнению заданий, шкалы и критерии оценивания

Критерии оценки:

оценка «отлично» выставляется обучающемуся, если правильный ответ дан не менее чем на 75% вопросов;

оценка «хорошо» выставляется обучающемуся, если правильный ответ дан не менее чем на 50% вопросов;

оценка «удовлетворительно» выставляется обучающемуся, если правильный ответ дан не менее чем на 30% вопросов;

оценка «неудовлетворительно» выставляется обучающемуся, если правильный ответ дан менее чем на 30% вопросов.

20.3 Фонд оценочных средств сформированности компетенций студентов, рекомендуемый для проведения диагностических работ

ПК-2 Непосредственное руководство процессами разработки программного обеспечения
ПК-3 Управление аналитическими работами

1) закрытые задания (тестовые, средний уровень сложности):

Вопрос 5

Установите соответствие, используя знания о паттерне MVVM.			*MAT*
#	**Вопрос**	**Ответ**	
1.	View	Интерфейс пользователя	
2.	Model	Данные и логика приложения	
3.	ViewModel	Интерпретация данных и логики для представления	

Вопрос 1

Что из перечисленного ниже относится к структурам в Swift?			*MA*
#	**Ответы**	**Отзыв**	**Оценка**
A.	могут иметь хранимые переменные		*25*
B.	могут иметь вычисляемые переменные		*25*
C.	могут содержать функции		*25*
D.	передаются по ссылке		*0*
E.	передаются по значению		*25*
F.	могут наследовать		*0*

Вопрос 10

B Swift можно возвращать несколько значений из функции, используя...			*MC*
#	**Ответы**	**Отзыв**	**Оценка**
A.	Кортеж		*100*
B.	Как массив, так и кортеж		*0*
C.	Массив		*0*
D.	Ни один из вышеперечисленных		*0*
	Общий отзыв к вопросу:		
	Для любого правильного ответа:	Ваш ответ верный.	
	Для любого неправильного ответа:	Ваш ответ неправильный.	
	Подсказка 1:		
	Показать количество правильных ответов (Подсказка 1):	Нет	
	Удалить некорректные ответы (Подсказка 1):	Нет	
	Теги:		
Позволяет выбирать один или несколько правильных ответов из заданного списка. (MC/MA)			

ФИИТ_маг ПК-6 Пр на iOS (ВО) Вопрос 11

Как можно добавить хранение в протокол в Swift?	*MC*

Балл по умолчанию:	1
Случайный порядок ответов	Да
Нумеровать варианты ответов?	0
Штраф за каждую неправильную попытку:	33.3
ID-номер:	

#	Ответы	Отзыв	Оценка
A.	С помощью расширения		*0*
B.	Реализовать методы		*0*
C.	Никак		*100*

Вопрос 12

Какой тип имеет переменная? Код представлен на языке Swift. var number: Int?	*MC*

#	Ответы	Отзыв	Оценка
A.	Int		*0*
B.	Optional		*100*
C.	None		*0*

Вопрос 2

Что из перечисленного ниже относится к классам в Swift?	*MA*

#	Ответы	Отзыв	Оценка
A.	могут иметь хранимые переменные		*20*
B.	могут иметь вычисляемые переменные		*20*
C.	могут содержать функции		*20*
D.	передаются по ссылке		*20*
E.	передаются по значению		*0*
F.	могут наследовать		*20*

Вопрос 3

Что из перечисленного ниже относится к протоколам (в т.ч. протоколам с расширениями) в Swift?	*MA*

#	Ответы	Отзыв	Оценка

Что из перечисленного ниже относится к протоколам (в т.ч. протоколам с расширениями) в Swift?			*MA*
#	**Ответы**	**Отзыв**	**Оценка**
A.	могут иметь хранимые переменные		*0*
B.	могут иметь вычисляемые переменные		*33.3*
C.	могут содержать функции		*33.3*
D.	передаются по ссылке		*0*
E.	передаются по значению		*0*
F.	могут наследовать		*33.3*

Вопрос 4

Что из перечисленного ниже относится к перечислениям в Swift?			*MA*
#	**Ответы**	**Отзыв**	**Оценка**
A.	могут иметь хранимые переменные		*0*
B.	могут иметь вычисляемые переменные		*33.3*
C.	могут содержать функции		*33.3*
D.	передаются по ссылке		*0*
E.	передаются по значению		*33.3*
F.	могут наследовать		*0*

Вопрос 6

Что из перечисленного НЕ относится к value-типам в Swift?			*MA*
#	**Ответы**	**Отзыв**	**Оценка**
A.	Перечисления		*0*
B.	Структуры		*0*
C.	Функции		*50*
D.	Классы		*50*

Вопрос 7

Что из перечисленного относится к value-типам в Swift?			*MA*
#	**Ответы**	**Отзыв**	**Оценка**

Что из перечисленного относится к value-типам в Swift?			*MA*
#	**Ответы**	**Отзыв**	**Оценка**
A.	Перечисления		*50*
B.	Структуры		*50*
C.	Функции		*0*
D.	Классы		*0*

Вопрос 8

Что из перечисленного НЕ относится к reference-типам в Swift?			*MA*
#	**Ответы**	**Отзыв**	**Оценка**
A.	Перечисления		*50*
B.	Структуры		*50*
C.	Функции		*0*
D.	Классы		*0*

Вопрос 9

Что из перечисленного относится к reference -типам в Swift?			*MA*
#	**Ответы**	**Отзыв**	**Оценка**
A.	Перечисления		*0*
B.	Структуры		*0*
C.	Функции		*50*
D.	Классы		*50*

2) открытые задания (тестовые, средний уровень сложности):

Вопрос 1

Опишите парадигму конструирования MVVM.		
	Шаблон ответа	**Информация для оценивающих**
		MVVM — это модель организации кода. То есть - определение мест, где “живет” код приложения. Model — UI НЕзависима. Она включает в себя данные и логику работы приложения. View - пользовательский интерфейс, отражает Model. Данные всегда поступают ОТ Model к View. Т. е. View отражает то, что находится в Model. Работа ViewModel состоит в том, чтобы “привязать” View к Model. Как только происходят хоть какие-то изменения в Model, View тут же отражает эти изменения. Т. е. ViewModel может интерпретировать Model для View.

Вопрос 2

Приведите пример любого протокола на Swift с одной функцией и одной переменной. Опишите структуру, объявляющую о реализации приведенного протокола.		
	Шаблон ответа	**Информация для оценивающих**

		protocol MyProtocol {
		func function (x: Int)
		var data: Int { get }
		
		struct structure: MyProtocol {
		// должен реализовать здесь function (x:) и data
		}

Вопрос 3

В чем состоит суть реактивного UI в SwiftUI?		
	Шаблон ответа	**Информация для оценивающих**
		Суть реактивного UI заключается в том, что когда происходят изменения в Model, они мгновенно автоматически отображаются в View.

Вопрос 4

С помощью каких возможностей можно добавить к протоколу реализацию функции в Swift?		
	Шаблон ответа	**Информация для оценивающих**
		С помощью расширения протокола.

Вопрос 5

Что такое ключевое слово «some» в SwiftUI?

	Шаблон ответа	Информация для оценивающих
		Some - это непрозрачный тип результата. Можно понимать это, как обратный неявный универсальный заполнитель. То есть это способ вернуть тип без необходимости предоставлять подробную информацию о самом конкретном типе. Это ограничивает то, что вызывающая сторона должна знать о возвращаемом типе, предоставляя только информацию о его соответствии протоколу. Использование непрозрачного типа - это способ позволить компилятору решить, каким будет конкретный тип возвращаемой функции, на основе фактического возвращаемого значения, ограничивая параметры типами, которые соответствуют данному протоколу. Например, some View означает, что тело всегда будет реализовывать протокол View, но конкретный тип реализации не обязательно должен быть известен вызывающему.

ФИИТ_маг ПК-6 Пр на iOS (РО) Вопрос 6

Для чего используется GeometryReader в SwiftUI?		
	Шаблон ответа	**Информация для оценивающих**
		GeometryReader - это View со специальными свойствами. Он оборачивается вокруг того, что необходимо сделать адаптируемым к изменению размера. GeometryReader всегда принимает предложенное ему пространство и сообщает о его размере через переменную size. Т. е. он всегда знает, какой размер ему предложен, что позволяет регулировать то, как будут выглядеть все находящееся внутри GeometryReader.

1) закрытые задания (тестовые, средний уровень сложности):

1. Установите соответствие между типом очереди и описанием:
Главная очередь: Последовательная очередь – Он выполняется в основном потоке.
Глобальная очередь: Параллельная очередь – Он выполняется с разными приоритетами и совместно используется системой intire.
Пользовательская очередь: Последовательная/ параллельная очередь.

2. Установите соответствие между качеством обслуживания (Quality of Service) и типом задач?
User Interactive: работа, выполняемая в основном потоке, например анимация или операции рисования.
User Initiated: Работа, которую начинает пользователь и которая должна дать немедленные результаты. Эта работа должна быть завершена, чтобы пользователь мог продолжить.
Utility: Работа, которая может занять некоторое время и которую не нужно заканчивать сразу. Аналогично индикаторам выполнения и импорту данных.
Background: Эта работа не видна пользователю. Резервное копирование, синхронизация, индексация и т.д.

3. Выберите существующие в Swift уровни управления доступом:
А) private +
Б) private(set) +
В) fileprivate +
Г) private(get)
Д) open +
Е) close

4. Выберите НЕверные утверждения о SwiftUI и UIKit:

А) В UIKit пользовательские интерфейсы создаются с помощью конструктора интерфейсов перетаскивания.
Б) Приложения UIKit подключаются к коду с помощью outlets и actions.
В) В SwiftUI пользовательские интерфейсы создаются программно.
Г) SwiftUI доступен только в iOS 12.0 и более поздних версиях. +

5. Установите соответствие:
А) as - используется для восходящего кастинга
Б) as? - создает значение optional, возвращает значение nil в случае неуспешного кастинга
В) as! - не создает значение optional, создает значение указанного типа, программа завершается с ошибкой в случае неуспешного кастинга

6. Что такое Safe Area?
А) Safe Area позволяет создавать специальные ограничения, чтобы контент не был скрыт специальными аппаратными панелями iOS. +
Б) Safe Area - место, где можно писать потокобезопасный код.
В) Функции, которые не выбрасывают (throw) ошибок

7. Установите соответствие
Self - относится к любому типу, соответствующему протоколу.
self - относится к любому значению, которое содержит тип.

8. Выберите верные утверждения о mutating-функциях в Swift?
А) В Swift свойства типов значений по умолчанию могут быть изменены в его методах экземпляра.
Б) Нужно использовать ключевое слово mutating в методе экземпляра, чтобы изменить свойства типа значения. +
В) Mutating-функция имеет право изменять значения свойств. +
Г) Нужно использовать ключевое слово mutating в методе экземпляра, чтобы преобразовать value-тип в reference-тип.

9. Выберите верные утверждения о цепочках optional в Swift?
А) С помощью цепочек можно связать несколько запросов вместе. +
Б) Если какое-либо звено в цепочке равно нулю, то будет ошибка.
В) Optional цепочка возвращает значение, если вся цепочка завершается успешно. +

2) <u>открытые задания (тестовые, средний уровень сложности):</u>

1. Для чего используются контейнеры HStacks и VStacks?

Ответ: контейнеры позиционируют View внутри себя, распределяя пространство между своими сабвью. HStack располагает View горизонально, VStack - вертикально.

2. Перечислите основные отличия декларативной и императивной моделей проектирования пользовательского интерфейса.

Ответ: Императивный синтаксис использует операторы, которые изменяют состояние программы. Императивная программа состоит из команд, которые должен выполнять компьютер. Он фокусируется на описании того, как работает программа, путем реализации алгоритмов в явных шагах. С другой стороны, декларативный синтаксис использует желаемые результаты без явного перечисления команд, которые должны быть выполнены. Она фокусируется на том, что, а не на том, как. В императивном синтаксисе вам необходимо предоставить пошаговые инструкции для выполнения некоторого действия. В декларативном программировании вам нужно только описать действие, а не то, как его выполнить.

3. Для чего используется @State?

Ответ: @State - обертка свойства, которую можно использовать для обозначения состояния View. SwiftUI хранит ее в специальной внутренней памяти вне структуры View. К ней может получить доступ только связанный с ней View. Как только значение свойства @State меняется, SwiftUI перестраивает View для учета изменения состояния.

4. Дайте определение замыкания.

Ответ: Замыкание - это встроенная функция.

5. Что происходит в данном коде?

let s: String? = …
let ss = s ?? “hello”

Ответ: Оператор ?? используется для создания выражения, которое становится значением по умолчанию, если Optional не установлен. Если значение переменной s будет установлено, то в переменную ss запишется значение ассоциированных данных переменной s. Если значение переменной s не будет установлено (будет .none / nil), то в переменную ss запишется значение по умолчанию “hello”.