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

УТВЕРЖДАЮ

Заведующий кафедрой
Математического обеспечения ЭВМ
Абрамов Г.В.

22.03.2024г.

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ
Б1.В.01 Программирование на платформе iOS

1. Код и наименование направления подготовки/специальности:

02.04.02 Фундаментальная информатика и информационные технологии

2. Профиль подготовки/специализация: Технологии разработки мобильных приложений

3. Квалификация выпускника: магистр

4. Форма обучения: очная

5. Кафедра, отвечающая за реализацию дисциплины: МО ЭВМ

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(ы): 1**

9.**Цели и задачи учебной дисциплины**

Целями освоения дисциплины являются: изучение принципов обработки и анализа научно-технической информации в области мобильных разработок, получение навыков проведения исследований по отдельным вопросам области, изучение основ и получение практических навыков в области разработки программного обеспечения для мобильных устройств, а также - формирование в процессе обучения навыков проблемного мышления, умения видеть, формулировать проблемы и выбирать средства и способы их решения, а также - самостоятельно корректировать свою учебно-познавательную деятельность.

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, жизненным циклом разработки мобильных приложений;

изучение основных приёмов и методов программирования мобильных приложений;

знакомство с основными конструкциями соответствующего языка программирования;

получение практических навыков по разработке полноценного мобильного приложения с применением всех изученных принципов, методик, методов и средств;

умение планировать и решать поставленные задачи с учетом имеющихся ресурсов.

10. Место учебной дисциплины в структуре ООП: Дисциплина относится к части, формируемой участниками образовательных отношений, блока Б1. Изучение курса должно базироваться на знании обучающимися материала курса «Объектно-ориентированное программирование». Дисциплина является базовой для изучения курсов «Создание мобильных приложений iOS» и «Безопасность мобильных устройств».

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-1 ПК-2 ПК-6	Способен проводить работы по обработке и анализу научно-технической информации результатов исследований Способен осуществлять научное руководство проведением исследований по отдельным задачам Способен применять современные языки программирования, операционные системы, сетевые технологии, технологии тестирования в сфере разработки мобильных приложений	ПК-1.3 ПК-2.1 ПК-6.2	Выбирает методы решения поставленной задачи с учетом имеющихся ресурсов, а также теоретического обобщения научных данных, результатов экспериментов и наблюдений. Формирует план проведения научно-исследовательских работ Реализует методы решения прикладных задач в профессиональной сфере деятельности, владеет пакетами программного обеспечения, операционными системами, определяет	Знать: методы решения поставленной задачи с учетом имеющихся ресурсов, а также теоретического обобщения научных данных, результатов экспериментов и наблюдений Уметь: формировать план проведения научно-исследовательских работ, определять наиболее значимые критерии качества программного продукта, выделять оптимальный вариант. Владеть: методами решения прикладных задач в профессиональной сфере деятельности, пакетами программного обеспечения, операционными системами.

			наиболее значимые критерии качества программного продукта, выделяет оптимальный вариант.	

12. Объем дисциплины в зачетных единицах/час. — 5/180.

Форма промежуточной аттестации: экзамен

13. Трудоемкость по видам учебной работы

Вид учебной работы		Трудоемкость			
		Всего	По семестрам		
			1 семестр	-	-
Контактная работа					
в том числе:	лекции		32		
	практические				
	лабораторные		32		
	курсовая работа				
Самостоятельная работа			80		
Промежуточная аттестация		Экзамен	36		
Итого:			180		

13.1. Содержание дисциплины

п/п	Наименование раздела дисциплины	Содержание раздела дисциплины	Реализация раздела дисциплины с помощью онлайн-курса, ЭУМК
1. Лекции			
1.1	Логика курса и введение в SwiftUI	Разработка приложений для iOS Использование SwiftUI Язык программирования Swift Функциональное программирование “Реактивная” концепция разработки пользовательского интерфейса (включая MVVM)	Программирование на платформе iOS
1.2	MVVM и система ТИПОВ в Swift.	Знакомство с парадигмой конструирования MVVM Структуры, классы, протоколы, перечисления, функции	Программирование на платформе iOS
1.3	Реактивный UI. Протоколы Protocols. Layout.	Реактивный пользовательский интерфейс (UI). Протоколы protocol и их комбинация с Generic — мощная сила в Swift. Система Layout (управление расположением Views) в SwiftUI.	Программирование на платформе iOS
1.4	Grid. Перечисления Enum. Optionals.	Создание пользовательского контейнера Grid и его применение к карточной игре Memorize	Программирование на платформе iOS

		Возможности перечислений enum в Swift Цель применения Optionals	
1.5	ViewBuilder. Shape. ViewModifier.	Управление доступом (Access Control). Больше о рисовании, включая конструирование @ViewBuilder для условного списка Views, протокол protocol Shape для нестандартного рисования и ViewModifier, механизм для создания пошаговых модификаций Views.	Программирование на платформе iOS
1.6	Animation.	Неявная и явная анимация, Анимация Views (через их модификаторы ViewModifiers, которые реализуют Animatable протокол) Transitions (анимируют появление / исчезновение Views с помощью определенных ViewModifiers) Анимация Shapes (через Animatable протокол)	Программирование на платформе iOS
2. Лабораторные работы			
2.1	Введение в Swift	Создание приложения для iOS (карточный игры на совпадение под названием Memorize)	Программирование на платформе iOS
2.2	MVVM	Разработка демонстративного приложения Memorize продолжается с участием MVVM.	Программирование на платформе iOS
2.3	Optionals	Такая важная концепция языка программирования Swift, как Optionals, объясняется на демонстрационном примере Memorize по мере того, как полностью отрабатывается логика функционирования этой карточной игры.	Программирование на платформе iOS
2.4	Анимация	Анимация переворота карт, новая игра (new game) и и начисления бонусов при быстром поиске «совпадающих» карт.	Программирование на платформе iOS

13.2. Темы (разделы) дисциплины и виды занятий

№ п/п	Наименование темы (раздела) дисциплины	Виды занятий (часов)				
		Лекции	Практические	Лабораторные	Самостоятельная работа	Всего
1	Логика курса и введение в SwiftUI	4				4
2	MVVM и система ТИПОВ в Swift.	4			4	8
3	Реактивный UI. Протоколы Protocols. Layout.	8			4	12
4	Grid. Перечисления Enum. Optionals.	8			16	24
5	ViewBuilder. Shape. ViewModifier.	8			16	24
6	Введение в Swift			4	16	20
7	MVVM			4	8	12
8	Optionals			12	8	20
9	Анимация			12	8	20
	Итого:	32		32	80	144

14. Методические указания для обучающихся по освоению дисциплины

Указание наиболее сложных разделов, работа с конспектами лекций, презентационным материалом. При использовании дистанционных образовательных технологий и электронного обучения выполнять все

указания преподавателей по работе на LMS-платформе, своевременно подключаться к online-занятиям, соблюдать рекомендации по организации самостоятельной работы.

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	Ваттер, А. П. IPAD, IPHONE, MACBOOK и сервисы APPLE. Все о совместном использовании : руководство / А. П. Ваттер. — Санкт-Петербург : Наука и Техника, 2016. — 256 с. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/74666.
2	Вейн, Ч. Swift подробно / Ч. Вейн ; перевод с английского Д. А. Беликова. — Москва : ДМК Пресс, 2020. — 422 с. — ISBN 978-5-97060-780-0. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/131722.

Б) дополнительная литература:

№ п/п	Источник
1	Разработка и продажа программ для iPhone и iPad / Дмитрий Елисеев. – БХВ- Петербург, 2012. ISBN 978-5-9775-0687-8.
2	SWIFT: БАЗОВОЕ РУКОВОДСТВО [Электронный ресурс] / М. Цукалос // Linux Format (Линукс Формат) .— 2016 .— №4 .— С. 82-85 .— Режим доступа: https://rucont.ru/efd/566911

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

№ п/п	Ресурс
1	https://edu.vsu.ru/course/view.php?id=5156
2	https://cs193p.sites.stanford.edu Курс лекций Ipad and Iphone Application Development

* 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	Соколова В. В. Разработка мобильных приложений : учебное пособие. — Томск: Изд-во ТПУ, 2014. — 175 с.: ил.
2	Нахавандипур В. iOS. Приемы программирования. — Санкт- Петербург: Питер, 2014. — 832 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:

Аудитория должна быть оборудована учебной мебелью, компьютером, мультимедийным оборудованием (проектор, экран, средства звуковоспроизведения), допускается переносное оборудование.

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.	Логика курса и введение в SwiftUI	ПК-1 ПК-6	ПК-1.3 ПК-6.2	Собеседование
2.	MVVM и система ТИПОВ в Swift.	ПК-1	ПК-1.3	Собеседование
3.	Реактивный UI. Протоколы Protocols. Layout.	ПК-1 ПК-6	ПК-1.3 ПК-6.2	Лабораторная работа
4.	Grid. Перечисления Enum. Optionals.	ПК-6	ПК-6.2	Лабораторная работа
5.	ViewBuilder. Shape. ViewModifier.	ПК-2	ПК-2.1	Собеседование
6.	Введение в Swift	ПК-1 ПК-6	ПК-1.3 ПК-6.2	Лабораторная работа
7.	MVVM	ПК-6	ПК-6.2	Лабораторная работа
8.	Optionals	ПК-6	ПК-6.2	Лабораторная работа
9.	Анимация	ПК-2 ПК-6	ПК-2.1 ПК-6.2	Лабораторная работа, отчет
Промежуточная аттестация форма контроля - экзамен				Перечень вопросов Практическое задание

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, тесты с вариантами ответа (ПК-1.3; ПК-2.1; ПК-6.2)

Примеры лабораторных работ

Задание 1. Заставьте работать игру Memorize как демонстрировалась на Лекциях 1 и 2. Печатайте весь код, не пользуйтесь копированием и вставкой кода откуда-то.
Сейчас карты появляются в предсказуемом порядке (совпадающие карты всегда появляются рядом друг с другом, делая игру очень легкой). Перемешайте карты.
Наши карты в настоящий момент размещаются в один ряд (мы исправим это на следующей неделе). Это делает наши карты очень “высокими” и “худыми” (особенно в портретном режиме), что выглядит не очень

привлекательно. Заставьте карты иметь соотношение между шириной и высотой как 2/3 (что приведет к пустому пространству выше и ниже ваших карт, что вполне нормально.)
Пусть ваша игра начинается со случайного количества пар карт в диапазоне от 2-х пар карт до 5-ти пар карт.
Когда ваша игра случайным образом показывает 5 пар карт, шрифт, который мы используем для эмоджи будет слишком большим (в портретном режиме) и начнет обрезаться. Отрегулируйте шрифт для 5 пар карт (только), чтобы он был меньше .largeTitle. Продолжайте использовать .largeTitle, когда у вас 4 или меньше пар карт в игре.
Ваш UI должен работать в портретном или ландшафтном режимах на любых iOS устройствах. В ландшафтном режиме ваши карты будут больше (но по-прежнему с 2/3 соотношением сторон). Возможно, от вас вообще ничего не понадобится (это часть “силы” SwiftUI), но чтобы в этом убедиться поэкспериментируйте на различных симуляторах в Xcode.

Задание 2. Заставьте работать игру Memorize, которая демонстрировалась на Лекциях с 1 по 4. Печатайте весь код, не пользуйтесь копированием и вставкой кода откуда-то.
Ваша игра всё ещё должна перемешивать (shuffle) карты.
Введите в игру концепцию “Тема” (“theme”). Тема theme состоит из имени темы, множества используемых эмоджи, числа карт, предназначенных для показа на экране (которое по крайней мере в одной, но не во всех темах, должно быть случайно) и подходящего цвета для рисования (например, оранжевый цвет прекрасно подошел бы к теме Хэллоуина).
Ваша игра должна, по крайней мере, иметь 6 различных тем
Необходимо дать возможность добавлять новую тему одной строкой кода.
Добавьте кнопку “New Game”, которая заканчивает текущую игру и начинает новую. Эта новая игра должна иметь случайным образом выбранную тему. Эту кнопку вы можете разместить в любом месте на вашем UI, где вам кажется это более подходящим.
Покажите где-нибудь на вашем UI имя темы.
Формируйте счет в игре добавлением 2-х очков за каждое совпадение и штрафом в 1 очко за каждое несовпадение ранее увиденной карты.
Добавьте на ваш UI метку для счета в игре (score label).
Ваш UI должен работать как в портретном, так и в ландшафтном режимах на любом iOS устройстве. Карты могут иметь любое соотношение сторон (aspectRatio), какое вам нравится. Возможно, этот обязательный пункт Задания вообще не потребует от вас никакой работы (в этом и “сила” SwiftUI), но необходимо убедиться в этом, запуская приложение на различных симуляторах в Xcode.

Задание 3. Реализуйте игру Set в версии соло (для одного игрока).
Когда ваша игра запускается в первый раз, карты на короткое время не должны отображаться, но как только они появляются, необходимо немедленно сдать 12 карт, заставляя их "прилетать" из случайных мест за пределами экрана.

В процессе игры эффективно используйте все пространство на экране. Карты должны становиться меньше (или больше) по мере того, как на экране одновременно появляется больше (или меньше) места, при этом всегда используется столько места, сколько доступно, и карты должны быть "красиво расположены". Все это делает Grid, и вы можете им пользоваться. Все изменения расположения и / или размеров карточек должны быть анимированы.
Карты могут иметь любое соотношение сторон (aspect ratio), которое вам нравится, но все они должны всегда иметь одно и то же соотношение сторон (независимо от их размера и независимо от того, сколько карт отображается на экране одновременно). Другими словами, карты могут представляться перед пользователем большего и меньшего размера по мере того, как игра продолжается, но карты не могут "растягиваться" до других соотношений сторон (aspect ratio) во время игры.
Символы на картах должны быть пропорциональны размеру карты (т.е. большие карты должны иметь большие символы, а меньшие карты должны иметь меньшие символы).
Пользователи должны иметь возможность выбрать до 3 карт, прикоснувшись к ним, чтобы попытаться создать Set (т. е. 3 совпадающие карты (matching) в соответствии с правилами игры Set). Пользователю должно быть ясно видно, какие карты уже были выбраны.
После того, как были выбраны 3 карты (selected), вы должны показать, совпадают ли эти 3 карты (match) или нет (mismatch). Вы можете показать это как хотите (цветом, границами, фоном, анимацией, чем угодно).
Каждый раз, когда выбраны 3 карты (selected), пользователю должно быть ясно, совпадают (match) они или нет (mismatch) (и карты, входящие в не совпавшее трио, должны выглядеть иначе, чем карты, когда в случае выбора 1 или 2 карт).
Поддержите "отмену выбора" ("deselection"), путем повторного касания уже выбранных карт (но только если в данный момент выбраны 1 или 2 карты (не 3)).
Если вы касаетесь любой карты, когда уже выбраны 3 совпадающие (matching) карты, образующие Set, тогда …
согласно правилам игры Set, замените эти 3 совпавших (matching) Set карты новыми из колоды

совпадающие (matching) карты должны улетать (с анимацией) в случайные места за пределами экрана
заменяющие карты должны прилететь (с анимацией) из других случайных мест за пределами экрана (или из "колоды" где-нибудь на экране)
если колода пуста, то место, освобожденное совпадающими картами (которые не могут быть заменены), должно быть доступно для оставшихся карт (т.е. они, вероятно, станут больше)
если карта, которую вы коснулись, не была частью совпадающих (matching) карты, образующие Set, выберите эту карту
Когда выбрана новая карта и есть уже 3 выбранных (selected) и не совпавших карты, сделайте эти 3 не совпавших карты не выбранными (deselected), а новую карту выбранной (selected) (независимо от того, была ли она частью не совпавшего трио карт).
Вам необходимо также иметь кнопку "Deal 3 More Cards" (Сдай еще 3 карты) (согласно правилам игры Set).
при касании этой кнопки замените выбранные карты, если выбранные карты составляют Set (с анимацией прилета /улета, как описано выше)
или, если выбранные карты не составляют Set (или если выбрано менее 3 карт, в том числе и ни одной), организуйте прилет (т. е. анимируйте прибытие) 3 новых карты, чтобы присоединиться к уже имеющимся на экране (и не делайте их выбранными (selected))
Отключите эту кнопку, если колода пуста
У вас также должна быть кнопка "Новая игра", которая запускает новую игру (т. е. происходит возврат к 12 случайно выбранным картам). Карты также должны прилетать и улетать, когда это происходит.
Чтобы немного упростить себе жизнь, вы можете заменить "волнистую линию" ("squiggle") в игре Set на прямоугольник.
Вы должны создать свою собственную структуру Shape для ромба (diamond).
Еще одно облегчающее жизнь изменение заключается в том, что вы можете использовать полупрозрачный цвет для представления "штриховки" в качестве заливки ("striped" shading) символа. Обязательно выберите подходящий уровень прозрачности, который четко отличит "штриховку" от "закрашивания" ("solid") символа.
Вы можете использовать любые 3 цвета, если они четко отличаются друг от друга.
Вы должны использовать перечисление enum как значимую часть вашего решения..
Вы должны использовать замыкание (т.е. функцию в качестве аргумента) как значимую часть вашего решения.
Ваш пользовательский интерфейс должен работать в портретной или ландшафтной ориентации на любом устройстве iOS. Это, вероятно, не потребует от вас какой-либо работы (это часть мощи SwiftUI), но обязательно поэкспериментируйте с запуском на разных симуляторах в Xcode, чтобы быть уверенным.

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

Пример теста:
1. Ознакомьтесь с приведенным фрагментом документации и ответьте на вопрос: с помощью какого метода можно очистить словарь в Swift?

```
func removeAll(keepingCapacity: Bool)

Removes all key-value pairs from the dictionary.

func shuffled() -> [Self.Element]

Returns the elements of the sequence, shuffled.
func firstIndex(where: (Self.Element) throws -> Bool) rethrows -> Self.Index?
```

```
Returns the first index in which an element of the collection satisfies the
given predicate.

func mapValues<T>((Value) throws -> T) rethrows -> Dictionary<Key, T>

Returns a new dictionary containing the keys of this dictionary with the values
transformed by the given closure.
```

Правильный ответ: func removeAll(keepingCapacity: Bool)

2. Ознакомьтесь с приведенным фрагментом документации и ответьте на вопрос: какой метод относится к методам поиска в словаре в Swift?

func removeAll(keepingCapacity: Bool)

Removes all key-value pairs from the dictionary.

func shuffled() -> [Self.Element]

Returns the elements of the sequence, shuffled.
func firstIndex(where: (Self.Element) throws -> Bool) rethrows -> Self.Index?

Returns the first index in which an element of the collection satisfies the given predicate.

func mapValues<T>((Value) throws -> T) rethrows -> Dictionary<Key, T>

Returns a new dictionary containing the keys of this dictionary with the values transformed by the given closure.
Правильный ответ: func firstIndex(where: (Self.Element) throws -> Bool) rethrows -> Self.Index?

3. Зачем нужно изучать SwiftUI? Какие преимущества это дает?

Правильный ответ: Новый сдвиг парадигмы.

Разработка пользовательского интерфейса происходит очень быстро.

Поможет вам научиться отделять пользовательский интерфейс от бизнес-логики.

Поможет вам понять другие декларативные фреймворки, такие как ReactNative, Flutter, composite layout.

4. В чем разница между делегатами и обратными вызовами в iOS?

Правильный ответ: Делегирование просто означает, что вы передаете себя кому-то другому, когда хотите, чтобы этот кто-то уведомлял вас об изменениях, чтобы вы могли реагировать на них. Например, если ViewController взаимодействует с сетевой службой и хочет получать уведомления, когда эта служба выполнит какой-либо запрос, он сделает себя делегатом сетевой службы. Затем сетевая служба вызовет методы делегирования, когда это будет сделано.

Обратные вызовы аналогичны по функциям шаблону делегирования. Они делают то же самое: сообщают другим объектам, когда что-то произошло, и передают данные по кругу.

Что отличает их от шаблона делегирования, так это то, что вместо передачи ссылки на себя вы передаете функцию.

5. Как можно добавить хранение в протокол в Swift?

Правильный ответ: Никак

6. Какой тип имеет переменная? Код представлен на языке Swift.

var number: Int?

Правильный ответ: Optional

Описание технологии проведения

Текущая аттестация проводится на занятии одновременно во всей учебной группе в виде теста в электронной образовательной среде «Электронный университет ВГУ», адрес курса — https://edu.vsu.ru/course/ Тест составляется из материалов ФОСа, формируется системой автоматически путём добавления случайных вопросов, количество которых соответствует образцу билета. Большая часть вопросов проверяется автоматически, проверки преподавателем с ручным оцениванием требуют только отдельные вопросы, представленные в форме эссе. Ограничение по времени на каждую попытку — 30 минут, количество попыток — 1, выставление окончательной оценки — по высшему баллу.»

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

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

Задания раздела 20.1 пункт 2 рекомендуются к использованию при проведении диагностических работ с целью оценки остаточных знаний по результатам освоения данной дисциплины

20.2 Промежуточная аттестация

Промежуточная аттестация по дисциплине осуществляется с помощью следующих оценочных средств:

Собеседование по экзаменационным билетам (по билетам к зачету)

Пример экзаменационного билета

1. Нарисовать структуру типичного приложения для iOS в терминах MVC. На примере приложения Калькулятор показать и рассказать функции различных частей приложения.

2. Что делает следующий код?

```
If let calcVC = vc as? CalculatorViewController {
    let x = calcVC.displayValue
```

3. Напишите, для каких из трех структур данных (классы, структуры, перечисления) характерно:

а) могут иметь хранимые свойства
б) могут иметь вычисляемые свойства
в) могут иметь инициализаторы
г) могут наследовать
д) передаются по значению
е) передаются по ссылке

4. Какой тип будет иметь переменная x и почему? Приведите пример возможного использования переменной x.

```
let x = display?.text?.hashValue
```

Примечание. hashValue – это переменная var в String, которая имеет тип Int.

5. Что и когда произойдет при выполнении следующего кода?

```
lazy var brain = CalculatorBrain()
```

6. Какой тип segue нужно выбрать при использовании Split View Controller’a, чтобы наблюдать на экране одновременно оба MVC?

7. Рассмотрите следующий код

```
let array = [“One”, “Two”, “Three”]
array.append(“Four”)
```

Что произойдет в результате такого вызова?

8. В чем отличие следующих объявлений?

```
var someProperty: Int = 42 {
    willSet {…}
    didSet {…}
}

var anotherProperty: Double {
```

```
        get {…}
        set{…}
}
```
В каком случае используется каждый из вариантов и для чего?

9. Чем property frame отличается от property bounds у UIView? Приведите пример показывающий сходства и различия.

10. Когда вызывается func draw(_ rect: CGRect), и для чего и как он используется? Каковы особенности использования данного метода?

11. Что делают методы?
func setNeedsDisplay()
func setNeedsDisplay(_ rect: CGRect)
В каком случае они вызываются?

12. При инициализации контроллера из сториборда необходимо скорректировать размер UIView вызовом метода
self.customView.frame = CGRect(x:0, y:0, width:100, height:200);
В каком из следующих методов, вызываемых в процессе жизни контроллера, необходимо это сделать и почему?
func awakeFromNib()
func viewDidLoad()
func viewWillAppear(_ animated: Bool)
func viewDidAppear(_ animated: Bool)
func viewWillLayoutSubviews()
func viewDidLayoutSubviews()

Описание технологии проведения
Экзамен 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.