РЕКОМЕНДОВАНО НАЧИНАЮЩИМ iOS-РАЗРАБОТЧИКАМ

200+ практических задач

ЗАДАЧИ ПО ПРОГРАММИРОВАНИЮ НА ЯЗЫКЕ SWIFT 4.0

ВЛАДИСЛАВ САМОЙЛОВ

Содержание

О задачнике

Все задания сформированы и дополнены из курса «Профессия: ios-разработчик».

Уровень заданий: от элементарных, рассчитанных на тренировку и закрепление, до hard core.

Выполняя задания и проклиная автора:) - вы прокачиваетесь!

***Рекомендации к выполнению.** Старайтесь выполнять всё по пунктам, игнорируя стратегию: «сначала сделаю интересное, а потом - всё остальное».*

Обязательно читайте документацию перед выполнением задания.

Обратите внимание, что ключей к заданиям нет!

Это сделано по 2 причинам:

1. *Учитесь думать самостоятельно.*
2. *Учитесь делать запросы в Google.*

Удачи и успехов вам в решении всех задач!

1. Константы, переменные и типы данных

1) Напишите переменные и константы всех базовых типов данных: int, UInt, float, double, string. У чисел вывести их минимальные и максимальные значения.

2) Создайте список товаров с различными характеристиками (количество, название). Используйте typealias.

3) Напишите различные выражения с приведением типа.

4) Вычисления с операторами (умножение, деление, сложение, вычитание): создайте консольный калькулятор.

Результат вычислений должен быть похож на: «3 + 2 = 5».

5) С помощью readLine() потренируйтесь вводить информацию в консоли и выводить её с помощью print().

* **Примечание**: readLine() может не сработать в Playground.

Чтобы выполнить задание 4.1., следуйте алгоритму:
Открыть Xcode → Create a New Xcode Project → Платформа macOs → Command Line Tool.

2. Строки

1) Напишите с помощью строк своё резюме: имя, фамилия, возраст, где живете, хобби и т.п.

2) Соберите из этих строк 1 большую (вспоминаем интерполяцию) и выведите в консоль.

3) Напишите 10 строк, соберите их с помощью интерполяции и поставьте в нужных местах переносы на новую строку и пробелы (см. \n и \t).

4) Разбейте собственное имя на символы, перенося каждую букву на новую строку.

5) Создайте переменную типа Int и переменную типа String.

Тип Int преобразуйте в String и с помощью бинарного оператора сложите 2 переменные в одну строку.

ПРОДВИНУТЫЙ УРОВЕНЬ.

1) Создайте 5-6 строк с названиями городов. Затем создайте 5-6 строк с названиями стран.

Если название города совпадает со страной — выведите в консоль название страны, города и слово: ПРАВИЛЬНО!

Соответственно, если не совпало — название страны и города и слово НЕПРАВИЛЬНО.

2) Возьмите созданные страны и проверьте их: содержат ли они букву А(буква на ваш выбор).

3) Создайте строку — набор букв кириллицей. Буквы абсолютно любые, строчные.

4) Замените их, создав отдельную переменную, на заглавные. А потом на строчные латинские.

5) Посчитайте индекс под которым находится каждая отдельно взятая буква.

3. Массивы

1) Создайте массив учеников из 12 мальчиков

2) Добавьте 7 девочек: с помощью append и по определённому индексу: см. insert.
Выведите результат в консоль.

3) Удалите 7 учеников так, чтобы получилось 5 мальчиков и 5 девочек. См. Удаление по диапазону. Выведите результат в консоль.

4) Создайте массив оценок от 0 до 5 и второй — от 6 до 10. Объедените два массива в один.

5) Присвойте каждому ученику оценку. Так, чтобы в консоли получилось примерно: «Вася — 5 баллов.Неплохо.».

6) Создайте массив из чисел. Числа - это купюры в долларах. Посчитать количество денег и вывести в консоль.

ПРОДВИНУТЫЙ УРОВЕНЬ.

1) Создайте 10 строк с названиями стран. Создайте массив.

Проверьте строки: если в строке более 5 символов — добавить её в массив. Но если строка содержит букву А - заменить эту букву на другую и добавить изменённую строку в массив.

2) Создайте массив из 20-30 значений любого типа. Сделайте линейную сортировку (linear sort).

3) Сделайте сортировку пузырьком (bubble sort).

4) Сделайте быструю сортировку (quick sort).

5) Сделайте bubble sort немного быстрее — создайте алгоритм «шейкерной» сортировки, или коктейльный алгоритм.

6) Отсортируйте массив по алгоритму четно-нечетной сортировки.

4. Словари

1) Создайте 10 разных Dictionary с разными типами данных. Один из них должен содержать все месяца года на русском. Второй на английском.

2) Соберите все ключи и значения каждого Dictionary и распечатайте в консоль.

3) Создайте пустой Dictionary и через условный оператор if проверьте, пустой он или нет. Если пустой, то в условии добавьте в него пар значений.

4) Создайте словарь — ключ: «иномарка», «значение»: название машины.

5) Добавьте в него значение по ключу - «отечественная». Распечатайте ключи и значения в консоль. Затем отдельно только значения.
Удалите из словаря иномарку по ключу (двумя способами: через nil и removeValue().

ПРОДВИНУТЫЙ УРОВЕНЬ.

1) Создайте словарь, где ключ — фамилия солдата, а значение — его приветствие.

2) В цикле пройдитесь по всем ключам и распечатайте фамилию каждого солдата.

3) Сделайте тоже самое со значениями и распечатайте приветствие каждого солдата.

4) Отсортировать словарь так, чтобы фамилии шли по алфавиту.

5) Создайте логическую проверку: если ключ словаря — Иванов, то скажите, что это снайпер. Сделайте тоже самое со всеми ключами.

5. Кортежи

1) Создать кортеж с 3-5 значениями. Вывести их в консоль 3 способами.

2) Давайте представим, что мы сотрудник ГАИ и у нас есть какое-то количество нарушителей.

Задача. Создать кортеж с тремя параметрами:

- первый - превышение скорости: количество пойманных;
- второй - вождение нетрезвым: количество пойманных;
- третий - бесправники: количество пойманных.

Распечатайте наших бедокуров в консоль через print().

3) Выведите каждый параметр в консоль. Тремя способами.

4) Создайте второй кортеж — нашего напарника. Значения задайте другие.

5) Пусть напарники соревнуются: создайте третий кортеж, который содержит в себе разницу между первым и вторым.

Вывести в консоль тремя способами.

ПРОДВИНУТЫЙ УРОВЕНЬ.

1) Создать 10 кортежей-разработчиков, каждый с 4 параметрами: имя, возраст, опыт, специальность. Пока не задавайте никаких данных.

2) После создания всем задайте имя, 3 из них — задайте возраст (любые значения выше 23),4 — опыт и только 2 — специальность.

3) Используя проверки if/else, выводите сообщения по логике: если возраст больше 23, есть опыт и специальность это ios — в консоль выводите имя и сообщение- принят на работу.

4) Тех, кто младше 23, добавьте в массив и отсортируйте. Найдите в этом массиве максимальное значение и удалите его.

5) Создайте массив и в проверку из пункта 3 добавьте условие: если специальность не ios — добавить в созданный массив.

6. Управление потоком: циклы

1) Создайте массив "дни в месяцах":
Распечатайте элементы, содержащие количество дней в соответствующем месяце, используя цикл for и этот массив.

2) Создать в if и отдельно в switch программу которая будет смотреть на возраст человека и говорить куда ему идти в школу, в садик, в универ, на работу или на пенсию и тд.

3) В switch и отдельно в if создать систему оценивания школьников по 12 бальной системе и высказывать через print мнение.

4) Создать создайте цикл в цикле. В первом цикле интервал 200 раз во втором как доедем до 15 выйти со всех циклов.

ПРОДВИНУТЫЙ УРОВЕНЬ.

1) Сыграем в шахматы. У вас есть некоторое количество фигур на доске. Они расположены на определённых клетках. Ваша задача: делать ходы и сообщать об этом в консоль.

2) Если ход пересекается с координатой фигуры — сообщать об этом в консоль.

3) Если ход попадает на определённую координату — сообщать в консоль к примеру: «Шах» или «Мат».

Примечание. В качестве координат можно использовать кортежи. Не забываем пользоваться switch и циклами.

4) Если ход попадает на определённую координату — сделайте фигуру равной nil.

7. Функции

1) Создать 3 функции:

- первая функция принимает массив Int и сортирует его по порядку (возрастание). После этого распечатывает результат в консоль;
- вторая функция принимает массив String, меняет 1, 2 и 3 элементы и сортирует по алфавиту. После этого распечатывает результат в консоль;
- третья функция принимает 2 массива String и складывает их. После этого распечатывает результат в консоль.

2) Создать журнал для учителя, который будет принимать имя студента, профессию и оценку и записывает это все в массив. И внесите 10 студентов туда и распечатаете через цикл for.

3) Создать функцию которая принимает имя и фамилию, потом положить это в массив и вывести результат в консоль.

4) Создайте функцию которая принимает параметры и вычисляет площадь круга.

5) Создайте Dictionary с именем ученики , где ключ name и score, а значение (1 тюпл из 5 имен) и (второй тюпл из 5 оценок).И распечатайте только имена по ключу.

ПРОДВИНУТЫЙ УРОВЕНЬ.

1) Функция принимает 3 массива String. Задача: сложить их вместе, преобразовать в тип Int и посчитать сумму. Распечатать результат в консоль.

2) Создайте функцию, которая принимает строку-рассказ без пробелов и знаков препинаний, всё с маленькой буквы и на латинице.

3) Напишите код, который добавляет знаки препинания в строку-рассказ, переносит каждое новое предложение на новую строку.Добавьте заглавные буквы.

8. Замыкания

1) Написать, как понимаете замыкания: что это, для чего нужны?

2) Написать 10 своих замыканий на примере сортировок массивов.

3) Написать функцию, которая принимает массив, проверяет — пустой или нет. И если пустой — нужно записать туда значения.

4) Написать функцию - сайт который требует имя, фамилию, ник, емейл, пароль. Всё вывести в консоль.

5) Написать функции которые принимают в качестве аргументов массивы и словари и проверяют: пустые или нет. Если пустые — добавляют туда значения и выводят в консоль.

ПРОДВИНУТЫЙ УРОВЕНЬ.

1) Создайте функцию, которая принимает массив Double. Напишите алгоритм, который находит в массиве минимальное значение. Распечатайте результат в консоль.

2) Теперь в этом же массиве найдите максимальное значение. Распечатайте результат в консоль.

3) Создайте функцию, которая принимает массив букв (Characters). Отсортируйте массив так, чтобы гласные оказались в левой части, а согласные — в правой. Разделите массив на гласные и согласные (2 отдельных массива). Отсортируйте каждый по алфавиту. Распечатайте результат в консоль.

4) Сложите получившиеся массивы в 1. Распечатайте результат в консоль.

9. Перечисления

1) Напишите как понимаете enumerations:что это такое, в чем их смысл, зачем нужны. Ваше личное мнение: как и где их можно использовать?

2) Написать по 5-10 enum разных типов + создать как можно больше своих enumerations. Главное, соблюдайте правила написания: понятность и заглавная буква в начале названия. Пропустите их через switch и распечатайте (см.видео).

3) Создайте своё резюме с использованием enum: имя, фамилия, возраст, профессия, навыки, образование, хобби и т.д. - пункты на ваше усмотрение.

Пропустите ваше резюме сначала через if else, затем через switch — для того, чтобы было понимание читаемости и красоты кода.

Дайте свою оценку первому и второму варианту написания.

4) Представьте, что вы попали на завод Apple. Вам принесли MacBook, Iphone, Ipad, Apple Watch и сказали: «Раскрась их в разные цвета. Джони Айву нужно вдохновение».

Вы подвисли и начали раскрашивать. В итоге у вас получился красный MacBook, зеленый Ipad, розовый Iphone и буро-коричневый Apple Watch.

Инструкция: для цветов задаём через enumeration.

Наши девайсы располагаем в теле функции.

Итог программы: «Айфон — розовый».

10. Классы и структуры

1) Написать простые классы с наследованием и без.

2) Написать консольное приложение, в котором создать класс *House* и в нем несколько свойств - *width*, *height* и несколько методов - *build* (выводит на экран умножение ширины и высоты), *getWidth* и *getHeight* выводят на экран соответсвенно ширину и высоту.

3) Написать класс , а в нем метод который будет принимать букву (одну, может быть и типа string, просто будете передавать ему одну букву) и возвращать все имена которые начинаются на эту букву.

К примеру, А - Алексей, Александр, Артем, Антон и т. д. Внутри метода или полем класса (тут как удобно, не сильно критично) будет задаваться массив строк (string) в котором будут прописаны имена. Имена откуда-то подгружать не надо, их надо просто захардкодить. Метод должен возвращать отфильтрованный массив с именами.

Так же написать метод, который будет принимать массив строк как аргумент и выводить их всех на консоль с новой строки каждое имя. Им распечатаете на консоль то что у вас получилось отфильтровать.

4) Написать класс, который формирует массив учеников, сортирует и считает количество этих учеников. Если учеников больше чем 30, выводится сообщение типа «в школе нет мест».

5) Создать 5-10 своих структур.

6) Сделайте список покупок! Программа записывает продукты в массив. Если вызываем определённый продукт — в консоли пишем к примеру «Мёд — куплено!».

ПРОДВИНУТЫЙ УРОВЕНЬ.

1) Консольная игра «Кролики в бочке».
Условия:

а) кроликов должно быть не меньше 9, равно как и бочек;
б) каждый кролик и бочка — имеют свой цвет;
в) бочка также имеет свой порядковый номер.

Задача: если мы выбираем красную бочку 2 — фиолетовый кролик выбран и выпрыгивает из бочки. Нужно добавить его в отдельный массив.

2) С помощью функции и кортежа (tuple) двигайте наших кроликов из бочки в бочку. Результат в консоли: красный кролик из розовой бочки № 1 попал в черную бочку № 4.

11. Свойства экземпляра

1. Создайте 10-20 классов с разными свойствами: хранения, вычисляемые (запись и чтение, просто запись, просто чтение), ленивые, обязательно потренируйтесь с observers (Наблюдатели).

2. Напишите класс Калькулятор. Создать вычисляемые свойства, которые складывают, умножают, делят, вычитают. И выведите результаты в консоль(распечатайте).

3.Создайте свой AppStore! Чтобы можно было записывать и сохранять, а потом вызывать новые приложения в каждой категории. Результаты — в консоль.

4. Добавьте своему AppStore отзывы. К каждому приложению. В виде текстовых сообщений.

5. Добавьте возможность поставить оценку приложению. Но не более 5 баллов. Если кто-то пытается вставить 6 или выше — какое-нибудь сообщение.

6. Подумайте, как можно добавить возможность удаления приложений из вашего AppStore. Напишите в коде.

7. Написать 5-10 разных функций, глобальных и вложенных.

8. Написать 5-10 сортировок массивов и словарей, с замыканиями и без.

ПРОДВИНУТЫЙ УРОВЕНЬ.

9. Поиграем в войнушку, как в детстве?:)

Создайте 3 класса: Орк, Человек, Эльф.

У человека есть свойства (задайте на своё усмотрение), например, weaponDamage и health и функция, которая рассказывает об уроне от оружия.

Эльф и Орк — наследники Человека. У Орка должен быть Наблюдатель, который сообщает об изменении health.

У Эльфа — тоже Наблюдатель, который сообщает об изменении weaponDamage (вроде:

Оружие Эльфа стало наносить больший урон! Нужно купить броню мощнее).*

12. Свойства типа

1. Написать, что такое глобальные и локальные переменные (своими словами, как поняли) и что такое свойства типа (в чем отличие от свойств экземпляра?)

2. Создать глобальный массив значений (тип на ваше усмотрение).

Создать 2 класса:

- первый класс сортирует массив по индексам с первого до последнего;
- второй класс сортирует массив по индексам с последнего до первого.

Вывести в консоль.

3. Создать свою компанию. В ней 2 класса-руководителя: генеральный директор и зам. по продажам.

У генерального 4 свойства. Сделать их разными: свойства типа и свойства экземпляра.

У зама по продажам (а он класс-наследник генерального) свойства наследуются + есть ещё 2 своих: количество продаж и марка машины, к примеру. Все значения свойств вывести в консоль.

И третий класс — Бухгалтер. У него есть пустой метод, который принимает глобальный переменный массив Работа (тип Int) и плюсует все элементы, получая сумму всех значений в массиве.
Сумму вывести в консоль.

4. Создайте класс «Монстр», который будет содержать имя, породу, с какой планеты прилетел, рост, вес, оружие.

Добавьте несколько свойств класса:

- минимальный рост и максимальный вес монстра;
- минимальную и максимальную мощность оружия.

Создайте свойство, которое будет содержать количество созданных экземпляров этого класса.

4.1. Создайте наблюдателя, который будет отслеживать наносимый ущерб от оружия монстра.

13. Методы

1) Создайте класс Программист.

Задайте следующие свойства:

a) язык программирования,
b) библиотеки, которые программист знает,
c) паттерны, которые программист знает,
d) английский язык,
e) наличие портфолио.

2) Напишите метод который будет распечатывать эти свойства в консоль.

3) Создайте класс HRManager.

4) Напишите метод, который будет оценивать знания программиста и решать: брать на работу или нет.
Результат в консоли: «Не знает что такое MVC.Надо подумать.» и т. п.

ПРОДВИНУТЫЙ УРОВЕНЬ.

1) Создайте наблюдателя, который отслеживает: прошёл ли программист собеседование. И если нет — добавьте этого программиста в массив.

14. Индексы

1) Создайте класс Карта. Задайте ему набор координат. Задайте каждой координате название. К примеру: горы, река, замок, пустошь, неизвестная местность.

2) Задайте каждой координате окраску.

3) Создайте класс персонажа, который будет двигаться по координатам.

4) Создайте проверку: если персонаж попал, к примеру, в красные горы, то выводите сообщение на консоль.

ПРОДВИНУТЫЙ УРОВЕНЬ.

1) Создайте своё войско, которые будут располагаться каждый на определённой координате.

2) Создайте наблюдателей, которые будут отслеживать: если персонаж перешёл на клетку,

которая занята другим персонажем — выводите сообщение, например: «Орки объявили войну Эльфам».

3) Добавьте алгоритм, который отслеживает чтобы нельзя было занять чужую координату.

4) Добавьте метод, который позволяет персонажам обмениваться местами.

5) Добавьте персонажам возможность наносить удары по координатам. Если есть попадание: выводите соответствующее сообщение в консоль и удаляйте персонажа.

6) Добавьте возможность соревнования.

15. ARC

1) Создайте класс Охранник. Задайте ему свойство: посетитель. Создайте несколько экземпляров, которые не имеют ссылок на класс Охранник.

2) Вызовите у Охранника конструктор (init) и получите сообщение в консоль о том, что посетитель по имени «Иван» пришёл.

3) Сделайте так, чтобы посетитель ушёл, вызвав деконструктор(deinit).

4) Есть Компания, у неё есть директор, зам.директора и менеджеры по продажам.

Пояснение: Директор — родительский класс, зам.директора его наследник, менеджеры — наследники зам.директора.

5) Менеджеры по продажам должны посылать друг другу сообщения. Вроде: «как продажи?» или «Передай мне отчет».

6) Реализуйте возможность у Менеджеров спрашивать Зам. директора про зарплату.

7) Реализуйте Зам. директору возможность отправлять отчет Директору.

ПРОДВИНУТЫЙ УРОВЕНЬ.

1) Компания может выводить в консоль список всех сотрудников.

2) Директор может обращаться к Компании и получить список всех сотрудников.

3) Реализуйте возможность уволить сотрудника (см.deinit).

16. ООП Парадигма

1) Создайте класс IT-компания. У него есть свойства: имя, должность, опыт, зарплата. И метод: workHard.

Соответственно, экземпляры вашего класса должны каждый делать что-то своё: программист кодит, дизайнер создаёт дизайны, тестировщик — тестирует, проект-менеджер общается с клиентами.

Создайте методы writeCode, testABug, createAdesign,CallToClients.
Сообщаем их действия в консоль.

2) Создайте классу свойство bugs типа Int. Создайте методы CatchBugs, FixBugs.

3) Тестировщик вызывает метод CatchBugs, который прибавляет 1 к свойству bugs.

4) Программист вызывает метод CreateBugs, который отнимает от bugs 1 каждый раз при вызове.

5) Отследите момент, когда багов стало больше и пусть тестировщик скажет: отправил на исправление.

6) Отследите момент когда багов стало меньше и сделайте соответствующее объявление от программиста.

7) Создайте класс-наследник IT-компании.Назовите его Designers. Создайте ему ему массив prototypes. Сделать проверку: если он пустой, добавлять туда значения. В методе createAdesign.

8) Если в prototypes значений стало больше чем 10 — сделать сообщение: «Дизайн готов.Передаю в работу программисту».

9) Каждый раз, когда вызывается метод writeCode — количество значений в массиве prototypes уменьшается на 1.

10) После того, как из prototypes уберёте все значения — программист выводит сообщение: «Проект готов. Отправляю тестировщику».

11) Добавьте дизайнеру следующее: когда пытаетесь присвоить ему имя, он его постоянно меняет на какое-то своё.

17. Опциональные цепочки

1) Создать класс Животное. Ему назначьте свойства: собака, кошка, свинья, корова, курица, лев, ягуар и т.п.
Пусть их будет не меньше 20.

2) Каждому животному задайте тип: домашнее, дикое, опасное.

3) Создайте метод, который фильтрует животных по типу и добавляет их в определённый массив.

4) Создайте класс Фермер. Ему назначьте свойства: имя, должность, зарплата.

5) Создайте 10-20 фермеров. Каждому назначьте разные имена, должности и зарплаты. Посчитайте, кто из них получает наибольшую зарплату.

6) С помощью перечислений распределите их по половому признаку. Создайте метод, который фильтрует женщин и мужчин и добавляет в разные массивы.
Массивы распечатать в консоль.

7) У мужчин вызывать метод hardWork, у женщин — homeWork.
HardWork будет просто говорить: «работаю в поле», homeWork - «Убираю в доме».

ПРОДВИНУТЫЙ УРОВЕНЬ.

8) Создайте класс Охотник. У него создайте метод, который будет искать опасных и диких животных (см. Задание 1.2) и «убивать» их.

Результат в консоли: «Охотник убил змею».Соответственно, количество элементов массива сокращается.

9) Отследите: если массив с дикими животными больше не содержит значений, добавляйте в массив с опасными животными новые значения.

10) Создайте Охотнику свойства: имя и специальность. К примеру, змеелов. Создайте 5-7 таких охотников, разных специальностей. Распечатайте в консоль.

11) Создайте класс Ферма. Ему задайте метод, который будет объединять всех: и животных, и фермеров, и охотников. Добавьте их в один массив и отсортируйте по алфавиту.

18. Обработка ошибок

1) Создайте класс Системный Администратор. Задайте ему свойства: Вирус, Нет сети, ddos-атака.

2) Создайте метод, который будет ловить ошибки. Сделайте так, чтобы был выход из метода раньше, чем будут «пойманы» ошибки.

3) Сделайте запрет на передачу ошибок.

ПРОДВИНУТЫЙ УРОВЕНЬ.

1) Если ваш метод не поймал вирус (вирус = nil) — сделайте вирус опциональным. То же самое проделайте с другими свойствами.

19. Проверка и приведение типов

1) Создайте класс Планета. Задайте ему 3 свойства: название, размер, цвет.

2) Создайте класс Марс. Сделайте его наследников класса Планета. Скройте(инкапсулируйте) его цвет и добавьте новое свойство: население.

3) Марсу сделайте метод changeSize. С таким условием, что каждый раз когда этот метод вызывается — размер Марса увеличивается в 2 раза.

4) Создайте класс Сатурн. Добавьте ему свойство Кольца типа Bool. Выполните условие: когда размер планеты превышает 30, кольца видны, если меньше 10 — кольца не видны.

5) Создайте 5-6 экземпляров каждого класса. Добавьте их в массив и проверьте: какого типа все эти экземпляра.

ПРОДВИНУТЫЙ УРОВЕНЬ.

1) Создайте класс Темная планета. Сделайте так, чтобы он уничтожал все другие планеты. **Примечание:** вспоминаем ARC

.

2) Создайте класс Марсианин. Сделайте его наследником класса Марс. Создайте ему свойство humanoid.

3) Сделайте проверку типа: если он принадлежит классу Марс, то прибавьте к humanoid 1 и добавьте в массив.

20. Расширения

1) Создайте класс Трансформер. Задайте ему свойства: имя, статус (автобот или десептикон), текущий полученный урон, количество урона, который может нанести (задайте по умолчанию 0).

2) Создайте ему метод стрельбы, который прибавляет к количеству урона 1.

3) Создайте 8-10 разных трансформеров. Выведите их имена, статус и количество наносимого урона в консоль.У некоторых вызовите метод стрельбы.

4) Создайте класс Кибертрон. В нём сформируйте коллекцию из ваших трансформеров.

5) Сделайте расширение класса Трансформер. Добавьте метод totalDestruction, который возводит количество наносимого урона в куб.

6) Сделайте расширение класса Кибертрон. Отсортируйте в нём коллекцию трансформеров так, чтобы сначала шли все автоботы, а потом десептиконы.

7) Вынесите в отдельные коллекции десептиконов и автоботов.

8) Поставьте проверку в Кибертрон так, чтобы вы смогли отследить когда десептиконов стало больше. И если это произошло — убирайте из коллекции 1 элемент.

9) С помощью сабскриптов поменяйте имя и статус любому трансформеру.

21. Протоколы

1) Создайте протокол Собеседование. У него есть свойство: фамилия. И методы: successOfinterview и failureOfInterview.

2) Создайте класс Программист. У него задайте свойства: skill типа String и inSearchOfJob типа Bool. Сделайте их опциональными. Программисту подключите протокол Собеседование.

3) Добавьте 2 массива, в которые будете добавлять программистов. Вроде okArray и failArray.

4) В методе successOfinterview сделайте проверку свойства inSearchOfJob. Если true — выводите сообщение «Подходит» и добавляйте имя в массив + сортируйте его по алфавиту.

5) В методе failureOfInterview просто выводите сообщение: «Не подходит».

6) Добавьте наблюдателя, который будет отслеживать как добавляются подходящие программисты.

7) Добавьте протокол Техническое собеседование (наследник Собеседования). Добавьте ему свойство isSuitable типа Bool и методы: successTechnicalOfinterview и failureTechnicalOfInterview.

8) successTechnicalOfinterview проверяет inSearchOfJob и skill: если inSearchOfJob == true, а skill == «ios», то добавляет этого программиста в массив okArray. И выводит сообщение об успешно пройденном собеседовании. «Вы нам подходите. Ждите оффер».

9) failureTechnicalOfInterview просто выводит сообщение «Нам жаль, но программист(имя) не прошёл собеседование».

10) Создайте класс Тестировщик. Подключите ему протоколы Собеседование, Техническое собеседование. Также добавьте свойство exp(опыт) типа Int и имя типа String.

11) Сделайте простую проверку: если опыт больше 3 лет — вызвать метод successOfinterview, который распечатает в консоль: «Тестировщик Егор допущен к техническому собеседованию». А если меньше 3 лет - в методе failureOfInterview просто выводите сообщение: «Егор не подходит».

12) Добавьте тестировщику свойство skills и 2 массива, по аналогии с Программистом. Метод successTechnicalOfinterview проверяет skills: если skill == «QA», то добавляет этого тестировщика в массив okArray. И выводит сообщение об успешно пройденном собеседовании. «Вы нам подходите. Ждите оффер».

13) failureTechnicalOfInterview просто выводит сообщение «Нам жаль, но тестировщик (имя) не прошёл собеседование».

14) Создайте 10-15 тестировщиков и программистов.

ПРОДВИНУТЫЙ УРОВЕНЬ.

15) Создайте отдельный тип База Резюме. В нём объедините всех программистов и тестировщиков в общую коллекцию.

16) В массив failArray добавьте тех, кто не прошёл собеседование. Распределите их по отдельным массивам и найдите в каждом самое длинное имя.
Выведите его в консоль.

22. Дженерики

1) Создать 5 функций, каждая - с входящими параметрами универсальных типов. Преобразовать их параметры в Int, Double, Float, String, Character. Параметры внутри тела функции добавить параметры в массив и отсортировать.
Вывести результаты в консоль.

2) Создать класс, структуру, перечисление универсального типа — для тренировки.

3) Создать протокол Стрелять. У него создать метод shoot.

4) Создать класс Солдат. Ему задайте имя и звание. Подключите к классу протокол Стрелять. Класс должен быть с универсальным шаблоном. Вызовите у него метод протокола. Метод должен называть имя, звание.

5) Сделайте расширение класса Солдат. Добавьте свойство срок службы и наблюдайте за ним. Если срок службы больше 10 — прилетает сообщение «А вот и дембель!»,а если меньше — пусть чистит картошку.

6) Создайте несколько солдат с разным сроком службы. Добавьте их в массив. Отсортируйте и солдата с самым большим сроком удалите из массива.

23. Делегирование

1) Создайте 3 класса. Назовите их Apple, Google, Microsoft.
Пусть в каждом будет имя, количество товара, задолженность. И метод - itemEnded().

2) Создайте делегат — Завод. У него создайте методы: payOffDebt() и shipProduct().

3) Когда у класса нет задолженности и мало товара — завод отгружает новый товар.

4) Создайте второй завод-делегат, независимый. У него будет просто метод shipProduct(). Он не учитывает задолженность класса.

5) Сделайте так, чтобы Google и Microsoft брали товар у первого завода, а Apple — у второго. Создайте 5-6 экземпляров каждого класса.

6) Создайте третий завод-делегат.Также независимый. Задайте ему список производимых деталей: корпус, стекло, железо, камера и т.п.

7) Сделайте так, чтобы Apple покупала у него только стекло и камеры, Google — корпус,

Microsoft — железо и другие детали. И заявлять, что именно они покупают.

8) У третьего завода создайте метод: первый будет создавать отчет о продажах (говорить кто и что купил).

9) В классах создайте свойство: ComplainsAboutQuality. Сделайте так, чтобы каждый смог отправить жалобу своему делегату-заводу.

10) Создайте 10-12 жалобщиков. Сделайте так, чтобы Apple покупала у первого завода, Google — покупала только железо, а Microsoft — покупала у второго.

БОНУС. Продвинутые операторы

1) Создайте числа 1, 4, 15, 230, 65 — с помощью битовых чисел и преобразуйте в Int и Uint.

2) Инвертируйте числа с помощью NOT.

3) Создайте целое число 17 из двух бит двух чисел с помощью оператора AND.

4) Создайте 2 целых числа. Создайте новое число с помощью OR.

5) Создайте класс Home, у него задайте свойства типа Int. Высота и ширина. Перегрузите операторы +,-,*,/(плюс, минус, умножить, делить).

Optional: О курсе «Профессия ios-разработчик»

Что внутри.

4 блока обучения:

1) **SWIFT;**
2) **UIKit;**
3) **Паттерны;**
4) **Клиент-серверные приложения.**

Бонусный, бесплатный блок: возможность получить заказ на разработку мобильного приложения на ios. Прямой способ вернуть деньги за обучение.

Как проходит.

С нуля. Индивидуально.

Получаете доступ к 1 видео-уроку и ДЗ. Выполняете. Задаёте вопросы. ДЗ проверяется.

В случае верного выполнения — двигаемся к следующему уроку.

В конце каждого основного блока — своего рода контрольная. На проверку и закрепление знаний.