Оглавление

[**1.** **UMA-мультипроцессоры** 7](#_Toc103974107)

[**2.** **NUMA-мультипроцессоры** 7](#_Toc103974108)

[**3.** **Мультипроцессоры UMA с общей шиной** 7](#_Toc103974109)

[**4.** **Мультипроцессоры UMA с координатными коммутаторами** 8](#_Toc103974110)

[**5.** **Мультипроцессоры UMA с многоступенчатыми коммутаторными сетями** 8](#_Toc103974111)

[**6.** **Основным признаком PVP-систем является** 9](#_Toc103974112)

[**7.** **Много ЦПУ, общая периферия, одна ОС, локализованное расположение, связь через общую память** 9](#_Toc103974113)

[**8.** **Много ЦПУ, общая периферия, несколько ОС, локализованное расположение, связь по выделенным линиям** 9](#_Toc103974114)

[**9.** **Много ЦПУ, периферии, разные ОС, разное расположение, связь по сети** 9](#_Toc103974115)

[**10.** **Каждому ЦП - свою ОС. Такой способ организации ОС в мультипроцессорах имеет недостаток** 9](#_Toc103974116)

[**11.** **В модели "хозяин-подчиненный"** 10](#_Toc103974117)

[**12.** **Симметричные мультипроцессоры** 10](#_Toc103974118)

[**13.** **В распределенной ОС** 10](#_Toc103974119)

[**14.** **Серверная ОС** 10](#_Toc103974120)

[**15.** **Сеть с равноправными узлами называется** 11](#_Toc103974121)

[**16.** **Сеть, управляемая одним или несколькими серверами, называется** 11](#_Toc103974122)

[**17.** **Сети, сочетающие в себе качества одноранговых и сетей на основе сервера, называются** 11](#_Toc103974123)

[**18.** **Ethernet это** 11](#_Toc103974124)

[**19.** **Сетевая служба это** 11](#_Toc103974125)

[**20.** **К сетевым средствам относятся** 11](#_Toc103974126)

[**21.** **Гипервизор первого типа это** 11](#_Toc103974127)

[**22.** **Гипервизор второго типа это** 11](#_Toc103974128)

[**23.** **…aaS означает** 12](#_Toc103974129)

[**24.** **IaaS модель** 12](#_Toc103974130)

[**25.** **PaaS в сравнении с IaaS** 12](#_Toc103974131)

[**26.** **Основной недостаток SaaS** 12](#_Toc103974132)

[**27.** **Такой модели развертывания нет** 12](#_Toc103974133)

[**28.** **Частные облака** 13](#_Toc103974134)

[**29.** **Google Cloud или Microsoft Azure предоставляют** 13](#_Toc103974135)

[**30.** **IaaS — это** 13](#_Toc103974136)

[**31.** **PaaS — это** 13](#_Toc103974137)

[**32.** **SaaS — это** 13](#_Toc103974138)

[**33.** **К характеристикам облачных технологий не относится** 13](#_Toc103974139)

[**34.** **Arduino — это** 14](#_Toc103974140)

[**35.** **Industry 4.0 характеризуется** 14](#_Toc103974141)

[**36.** **IoT – это** 14](#_Toc103974142)

[**37.** **RFID-метка – это** 14](#_Toc103974143)

[**38.** **К предпосылкам развития Интернета вещей можно отнести** 14](#_Toc103974144)

[**39.** **К примерам проектов IoT можно отнести** 14](#_Toc103974145)

[**40.** **К типичным примерам применения технологии IoT относится** 15](#_Toc103974146)

[**41.** **Ключевая особенность технологии IoT – это** 15](#_Toc103974147)

[**42.** **Основной недостаток технологии IoT** 15](#_Toc103974148)

[**43.** **Технология IIoT – это** 15](#_Toc103974149)

[**44.** **К основным компонентам технологии IoT не относится** 15](#_Toc103974150)

[**45.** **Arduino Uno – это** 15](#_Toc103974151)

[**46.** **Устройства IoT получают данные от внешней среды с помощью** 16](#_Toc103974152)

[**47.** **Ключевая особенность микроконтроллера ESP8266 это** 16](#_Toc103974153)

[**48.** **Ассемблер - это частный случай** 16](#_Toc103974154)

[**49.** **Интерпретатор это** 16](#_Toc103974155)

[**50.** **Что такое препроцессор** 16](#_Toc103974156)

[**51.** **Какое утверждение является верным (вопрос об отношении между компилятором, транслятором и интерпретатором)** 17](#_Toc103974157)

[**52.** **Какое утверждение является неверным (вопрос об отношении между компилятором, транслятором и интерпретатором)** 17](#_Toc103974158)

[**53.** **На последнем этапе трансляции исходный код превращается в** 17](#_Toc103974159)

[**54.** **Исходный код это** 17](#_Toc103974160)

[**55.** **Для того, чтобы выполнить программу, необходимо** 17](#_Toc103974161)

[**56.** **Что из перечисленного не является этапом трансляции программы** 18](#_Toc103974162)

[**57.** **Компиляция это** 18](#_Toc103974163)

[**58.** **Программа, позволяющая управлять процессом выполнения программы и обнаруживать в ней ошибки** 18](#_Toc103974164)

[**59.** **gcc это** 19](#_Toc103974165)

[**60.** **Программа, которая собирает исполняемый файл из нескольких объектных модулей, называется** 19](#_Toc103974166)

[**61.** **К общим синтаксическим критериям не относится** 19](#_Toc103974167)

[**62.** **К функциям семантического анализатора не относится** 19](#_Toc103974168)

[**63.** **python это** 20](#_Toc103974169)

[**64.** **Необязательным этапом трансляции программы является** 20](#_Toc103974170)

[**65.** **Символ (буква) в формальных языках – это** 20](#_Toc103974171)

[**66.** **Алфавит в формальных языках – это** 20](#_Toc103974172)

[**67.** **Слова в формальных языках – это** 20](#_Toc103974173)

[**68.** **Все существующие формальные языки строятся на основе** 21](#_Toc103974174)

[**69.** **Цепочка символов (строка) – это** 21](#_Toc103974175)

[**70.** **Длиной цепочки называется** 21](#_Toc103974176)

[**71.** **В формальных языках и грамматиках цепочка ω называется подцепочкой γ, если γ=** 21](#_Toc103974177)

[**72.** **Язык L над алфавитом А: L(A) это некоторое счетное подмножество цепочек конечной длины из множества всех цепочек алфавита А: (in — является подмножеством)** 22](#_Toc103974178)

[**73.** **В формальных языках лексика это –** 22](#_Toc103974179)

[**74.** **В формальных языках лексема это –** 22](#_Toc103974180)

[**75.** **В формальных языках синтаксис это –** 23](#_Toc103974181)

[**76.** **В формальных языках семантика это –** 23](#_Toc103974182)

[**77.** **Для задания ЯП необходимо** 23](#_Toc103974183)

[**78.** **Какая из проблем не решается в теории формальных языков** 23](#_Toc103974184)

[**79.** **Что можно описать в виде строгого набора правил?** 23](#_Toc103974185)

[**80.** **Язык L над алфавитом А – это** 24](#_Toc103974186)

[**81.** **Формальная грамматика – это** 24](#_Toc103974187)

[**82.** **Форма Бэкуса-Наура используется для** 24](#_Toc103974188)

[**83.** **Распознаватель – это** 25](#_Toc103974189)

[**84.** **По типу считывающего устройства распознаватели разделяются на** 25](#_Toc103974190)

[**85.** **Выберите действие, которое не выполняется распознавателем** 25](#_Toc103974191)

[**86.** **К структурным компонентам распознавателя не относится** 26](#_Toc103974192)

[**87.** **Управляющее устройство называется детерминированным, если** 26](#_Toc103974193)

[**88.** **Конфигурация распознавателя называется начальной, если** 26](#_Toc103974194)

[**89.** **Формальная грамматика определяется с помощью** 26](#_Toc103974195)

[**90.** **К метасимволам языка РБНФ не относится** 26](#_Toc103974196)

[**91.** **Конфигурация распознавателя называется конечной, если** 27](#_Toc103974197)

[**92.** **Язык, определяемый распознавателем, это** 27](#_Toc103974198)

[**93.** **Грамматики по Хомскому типа 0 это –** 27](#_Toc103974199)

[**94.** **Закончите фразу: "Грамматики с фразовой структурой ..."** 27](#_Toc103974200)

[**95.** **Грамматики по Хомскому типа 1 это –** 27](#_Toc103974201)

[**96.** **Грамматики по Хомскому типа 2 это –** 27](#_Toc103974202)

[**97.** **Грамматики по Хомскому типа 3 это –** 28](#_Toc103974203)

[**98.** **Регулярные грамматики включают в себя** 28](#_Toc103974204)

[**99.** **Языки типа "0" по классификации Хомского это –** 28](#_Toc103974205)

[**100.** **Языки типа "1" по классификации Хомского это –** 28](#_Toc103974206)

[**101.** **Языки типа "2" по классификации Хомского это –** 28](#_Toc103974207)

[**102.** **Языки типа "3" по классификации Хомского это –** 29](#_Toc103974208)

[**103.** **Языки классифицируются в соответствии с(со)** 29](#_Toc103974209)

[**104.** **В случае, когда язык может быть задан с помощью большого количества грамматик, относящихся к разным типам, то из всех этих грамматик берется** 29](#_Toc103974210)

[**105.** **Определите тип языка, заданного правилами вывода: 1) S -> aaCFD; 2) AD -> D; 3) F -> AFB| AB; 4) Cb -> bC; 5) AB -> bBA; 6) CB -> C; 7) Ab -> bA; 8) bCD -> e, где e - цепочка нулевой длины** 29](#_Toc103974211)

[**106.** **Определите тип языка, заданного правилами вывода: 1) S -> At | Bt; 2) A -> a | Ba; 3) B -> b | Bb | Ab.** 30](#_Toc103974212)

[**107.** **Определите тип языка, заданного правилами вывода: 1) S -> APA; 2) P -> "+" | -; 3) A -> a | b.** 30](#_Toc103974213)

[**108.** **Определите тип языка, заданного правилами вывода: 1) S -> 1B; 2) B -> B0 | 1.** 30](#_Toc103974214)

[**109.** **Определите тип языка, заданного правилами вывода: 1) S -> aSBC | abC; 2) CB -> BC; 3) bB -> bb; 4) bC -> bc; 5) cC -> cc.** 30](#_Toc103974215)

[**110.** **Структурная единица языка, которая состоит из элементарных символов языка это** 30](#_Toc103974216)

[**111.** **Распознавание и выделение лексем из входной последовательности символов производится** 30](#_Toc103974217)

[**112.** **Лексический анализ** 31](#_Toc103974218)

[**113.** **Лексический анализ на лету – это** 31](#_Toc103974219)

[**114.** **Причинами, по которым в состав практически всех компиляторов включают лексический анализ являются** 31](#_Toc103974220)

[**115.** **В ЯП Pascal else является** 31](#_Toc103974221)

[**116.** **Функции, выполняемые лексическим анализатором и тип лексем, которые он должен выделять определяются** 31](#_Toc103974222)

[**117.** **Таблица (список) лексем содержит** 32](#_Toc103974223)

[**118.** **Если изменится кодировка исходной программы, то это повлияет** 32](#_Toc103974224)

[**119.** **К типам лексем ЯП не относится** 32](#_Toc103974225)

[**120.** **Теоретической моделью, используемой при построении лексического анализатора, является** 32](#_Toc103974226)

[**121.** **На этапе трансляции синтаксический анализ** 32](#_Toc103974227)

[**122.** **Синтаксический анализатор – это** 33](#_Toc103974228)

[**123.** **Что подается на вход синтаксического анализатора?** 33](#_Toc103974229)

[**124.** **Деревом вывода грамматики G(T,N,P,S) называется дерево (граф), которое соответствует некоторой цепочке вывода и удовлетворяет следующим условиям:** 33](#_Toc103974230)

[**125.** **МПА при построении синтаксического анализатора – это** 34](#_Toc103974231)

[**126.** **Одним из самых простых методов синтаксического анализа является** 34](#_Toc103974232)

[**127.** **Характерной особенностью автомата с магазинной памятью является** 34](#_Toc103974233)

[**128.** **Метод рекурсивного спуска применим к** 34](#_Toc103974234)

[**129.** **Синтаксис языков программирования описывают с помощью** 35](#_Toc103974235)

[**130.** **Детерминированный МПА** 35](#_Toc103974236)

[**131.** **Недетерминированный МПА** 35](#_Toc103974237)

[**132.** **Установление однозначного соответствия между лексическими единицами и их именами в тексте программы – это** 35](#_Toc103974238)

[**133.** **Ассемблер – это** 35](#_Toc103974239)

[**134.** **Что из перечисленного не является промежуточным языком** 36](#_Toc103974240)

[**135.** **Генерация кода** 36](#_Toc103974241)

[**136.** **Идентификация это** 36](#_Toc103974242)

[**137.** **Основная идея метода синтаксически управляемого перевода** 37](#_Toc103974243)

[**138.** **Назовите неправильное утверждение** 37](#_Toc103974244)

[**139.** **Дерево операций** 37](#_Toc103974245)

[**140.** **Для формирования внутреннего представления программы может применяться** 37](#_Toc103974246)

[**141.** **ПОЛИЗ — это** 38](#_Toc103974247)

[**142.** **Преимущество триад при промежуточном представлении программы перед тетрадами в том, что** 38](#_Toc103974248)

[**143.** **Выберите ПОЛИЗ для выражения (2+5) \* (5-8) / 3** 38](#_Toc103974249)

[**144.** **Выберите ПОЛИЗ для выражения (4-2) \* 4 / 5** 38](#_Toc103974250)

[**145.** **Промежуточный код** 38](#_Toc103974251)

[**146.** **К достоинствам триадного представления промежуточного кода не относится** 39](#_Toc103974252)

[**147.** **К недостаткам тетрадного представления промежуточного кода относится** 39](#_Toc103974253)

[**148.** **Линейный участок программы** 39](#_Toc103974254)

[**149.** **Для линейных участков не могут выполняться следующие виды оптимизации** 39](#_Toc103974255)

[**150.** **Что является критериями оптимизации кода** 39](#_Toc103974256)

[**151.** **Оптимизация это** 40](#_Toc103974257)

[**152.** **К приемам оптимизации линейных участков программы не относится** 40](#_Toc103974258)

[**153.** **К приемам оптимизации циклов не относится** 40](#_Toc103974259)

[**154.** **К приемам оптимизации не относится** 40](#_Toc103974260)

[**155.** **Особенностью оптимизации как этапа трансляции является** 41](#_Toc103974261)

[**156.** **BIOS используется** 41](#_Toc103974262)

[**157.** **MBR хранит** 41](#_Toc103974263)

[**158.** **MBR это** 41](#_Toc103974264)

[**159.** **Встроенное программное обеспечение** 41](#_Toc103974265)

[**160.** **Вторичный загрузчик** 42](#_Toc103974266)

[**161.** **Выберите истинное утверждение** 42](#_Toc103974267)

[**162.** **Досистемная загрузка происходит с помощью** 42](#_Toc103974268)

[**163.** **Задачей вторичного загрузчика является** 43](#_Toc103974269)

[**164.** **Задачей первичного загрузчика является** 43](#_Toc103974270)

[**165.** **Какое понятие относится к процессу нулевой загрузки** 43](#_Toc103974271)

[**166.** **Начальный этап загрузки зависит** 43](#_Toc103974272)

[**167.** **Нулевой этап загрузки является** 44](#_Toc103974273)

[**168.** **Работа вторичного загрузчика** 44](#_Toc103974274)

[**169.** **Что из перечисленного относится к вторичному загрузчику** 44](#_Toc103974275)

[**170.** **Что из перечисленного является загрузчиком Linux?** 44](#_Toc103974276)

[**171.** **Что такое BIOS** 45](#_Toc103974277)

[**172.** **Динамическая библиотека содержит** 45](#_Toc103974278)

[**173.** **Изменение динамической библиотеки, влияет на программы которые ее используют?** 45](#_Toc103974279)

[**174.** **Код статической библиотеки** 45](#_Toc103974280)

[**175.** **Отличительный признак динамической библиотеки** 46](#_Toc103974281)

[**176.** **При использовании статической библиотеки** 46](#_Toc103974282)

[**177.** **Процесс может использовать функции динамической библиотеки** 46](#_Toc103974283)

[**178.** **Расширение динамических библиотек в Linux** 46](#_Toc103974284)

[**179.** **Расширение статических библиотек в Linux** 47](#_Toc103974285)

[**180.** **Для указания импортируемых из DLL функций применяют конструкцию** 47](#_Toc103974286)

[**181.** **Для указания экспортируемых из DLL функций применяют конструкцию** 47](#_Toc103974287)

[**182.** **При загрузке DLL c неявной компоновкой** 47](#_Toc103974288)

[**183.** **При загрузке DLL c явной компоновкой** 47](#_Toc103974289)

[**184.** **Расширение динамических библиотек в Windows** 47](#_Toc103974290)

[**185.** **Расширение статических библиотек в Windows** 48](#_Toc103974291)

[**186.** **Функция входа в DLL — это функция** 48](#_Toc103974292)

[**187.** **К достоинствам использования динамических библиотек относят** 48](#_Toc103974293)

[**188.** **К недостаткам использования динамических библиотек относят** 48](#_Toc103974294)

[**189.** **Сравните файлы ls и ln, находящиеся в каталоге /bin, используя абсолютные имена файлов** 48](#_Toc103974295)

[**190.** **Вы находитесь в каталоге ~/. Просмотрите первые 10 строк файла /etc/group, используя относительное имя файла** 49](#_Toc103974296)

[**191.** **Вы находитесь в каталоге /var/opt. Просмотрите первые 10 строк файла /home/user/mytxt, используя относительное имя файла** 49](#_Toc103974297)

[**192.** **Вы находитесь в каталоге /opt/pycharm. Просмотрите первые 10 строк файла ~/mytxt, используя абсолютное имя файла** 49](#_Toc103974298)

[**193.** **Вы находитесь в каталоге ~. Где вы окажетесь после применения команды cd ../../../../../../../..** 49](#_Toc103974299)

[**194.** **Вы находитесь в каталоге ~. Где вы окажетесь после применения команды cd ../././././././vasia/./.././brother** 49](#_Toc103974300)

[**195.** **Вы находитесь в каталоге ~. Где вы окажетесь после применения команды cd 1/2/3/././4/././..** 49](#_Toc103974301)

[**196.** **Вы находитесь в каталоге /usr/local/bin. Где вы окажетесь после применения команды cd /../../../../etc/././..** 49](#_Toc103974302)

[**197.** **Вы находитесь в каталоге /usr/local/bin. Где вы окажетесь после применения команды cd /../var/./././** 50](#_Toc103974303)

[**198.** **Поменяйте владельца и группу файла ~/mytxt на root** 50](#_Toc103974304)

[**199.** **Установите, используя символы прав доступа следующие права на файл ~/mytxt: для себя все права, для группы и остальных — никаких** 50](#_Toc103974305)

[**200.** **Установите при помощи восьмеричных масок следующие права на файл ~/mytxt: для себя все права, для группы и остальных — никаких** 50](#_Toc103974306)

[**201.** **Установите при помощи восьмеричных масок следующие права на файл ~/mytxt: для себя исполнение и запись, для группы никаких, для остальных чтение** 50](#_Toc103974307)

[**202.** **Просмотрите руководство по команде more** 50](#_Toc103974308)

[**203.** **Создайте жесткую ссылку для файла /home/user1/mytxt с тем же именем, поместив ее в каталог /home/user1/work** 50](#_Toc103974309)

[**204.** **Создайте символьную ссылку для файла /home/user1/mytxt с тем же именем, поместив ее в каталог /home/user1/work** 51](#_Toc103974310)

[**205.** **Получите подробную информацию о файлах в текущем каталоге** 51](#_Toc103974311)

[**206.** **Получите список файлов текущего каталога, отсортированный по размеру** 51](#_Toc103974312)

[**207.** **Выведите на консоль имя самого большого (по размеру) файла текущего каталога** 51](#_Toc103974313)

[**208.** **Найдите все текстовые файлы (с расширением txt) текущего каталога, содержащие в названии "pro"** 51](#_Toc103974314)

[**209.** **Выведите список пользователей в алфавитном порядке** 51](#_Toc103974315)

[**210.** **Получите информацию о текущем пользователе** 51](#_Toc103974316)

[**211.** **Абсолютный загрузчик** 51](#_Toc103974317)

[**212.** **Динамический загрузчик выполняет связывание** 52](#_Toc103974318)

[**213.** **Для связывания модулей компоновщик использует** 52](#_Toc103974319)

[**214.** **Недостаток динамического загрузчика** 52](#_Toc103974320)

[**215.** **Преимущество динамического загрузчика** 52](#_Toc103974321)

[**216.** **Редактор связей** 53](#_Toc103974322)

[**217.** **Преимущество редактора связей** 53](#_Toc103974323)

[**218.** **Недостатки редактора связей** 53](#_Toc103974324)

[**219.** **Редактор связей выполняет связывание** 53](#_Toc103974325)

[**220.** **Связывающий загрузчик** 54](#_Toc103974326)

[**221.** **Связывающий загрузчик выполняет связывание** 54](#_Toc103974327)

[**222.** **Что не является преимуществом связывающего загрузчика** 54](#_Toc103974328)

[**223.** **Что является преимуществом связывающего загрузчика** 55](#_Toc103974329)

# **1. UMA-мультипроцессоры**

UMA – архитектура с однородным доступом к памяти.

В машинах UMA каждый процессор имеет одно и тоже время доступа к памяти.

# **2. NUMA-мультипроцессоры**

NUMA – архитектура с неоднородным доступом к памяти.

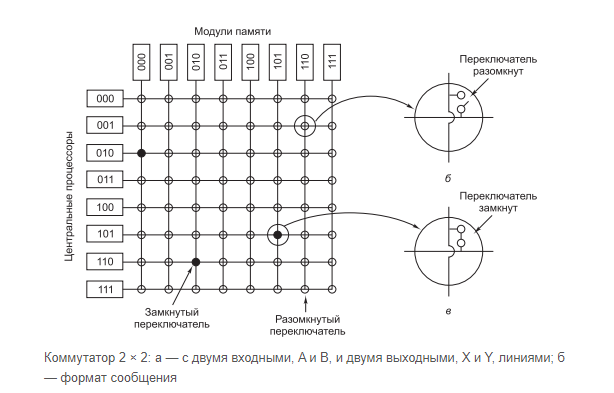
# **3. Мультипроцессоры UMA с общей шиной**

Простейшие мультипроцессоры основаны на использовании общей шины. Два и более центральных процессора и один и более модулей памяти используют для обмена данными одну и ту же шину.

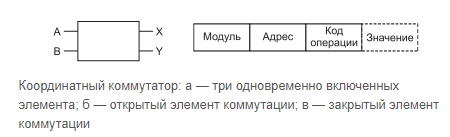


# **4. Мультипроцессоры UMA с координатными коммутаторами**

Простейшая схема подключения n центральных процессоров к k модулям памяти — это координатный коммутатор (crossbar switch). Координатный коммутатор десятилетиями использовался для коммутации телефонных переговоров, чтобы произвольным образом соединить группу входных линий с набором выходных линий.



# **5. Мультипроцессоры UMA с многоступенчатыми коммутаторными сетями**



На рисунке показана совершенно другая мультипроцессорная архитектура, построенная на простых коммутаторах 2 × 2. У такого коммутатора есть два входа и два выхода.

Сообщения, поступающие по любой входной линии, могут быть скоммутированы на любую выходную линию.

# **6. Основным признаком PVP-систем является**

Основным признаком PVP-систем является наличие специальных векторно-конвейерных процессоров, в которых предусмотрены команды однотипной обработки векторов независимых данных, эффективно выполняющиеся на конвейерных функциональных устройствах.

# **7. Много ЦПУ, общая периферия, одна ОС, локализованное расположение, связь через общую память**

Ответ: Мультипроцессор

# **8. Много ЦПУ, общая периферия, несколько ОС, локализованное расположение, связь по выделенным линиям**

Ответ: Многомашинная система

# **9. Много ЦПУ, периферии, разные ОС, разное расположение, связь по сети**

Ответ: Распределенная система

# **10. Каждому ЦП - свою ОС. Такой способ организации ОС в мультипроцессорах имеет недостаток**

Во-первых, когда процесс обращается к системному вызову, системный вызов перехватывается и обрабатывается его собственным центральным процессором при помощи структур данных в таблицах операционной системы.

Во-вторых, поскольку у каждой операционной системы есть свои собственные таблицы, у нее есть также и свой набор процессов, которые она сама планирует.

Совместного использования процессов нет. Один процессор может быть загружен, другой – простаивать. В-третьих, совместного использования страниц также нет. У одного ЦП много свободных страниц, другой будет постоянно заниматься свопингом. И нет никакого способа занять свободные страницы у соседнего процессора, так как выделение памяти статически фиксировано.

В-четвертых, и это хуже всего, если операционная система поддерживает буферный кэш недавно использованных дисковых блоков, то каждая операционная система будет выполнять это независимо от остальных. Таким образом, может случиться так, что некоторый блок диска будет присутствовать в нескольких буферах одновременно, причем в нескольких буферах сразу он может оказаться модифицированным, что приведет к порче данных на диске. Единственный способ избежать этого заключается в полном отказе от блочного кэша, что значительно снизит производительность системы.

# **11. В модели "хозяин-подчиненный"**

Модель «хозяин-подчиненный». Создается одна копия ОС и связывается с процессором – хозяином. Только этот процессор может выполнять системные задачи. Остальные процессоры (подчиненные) являются ресурсом для вычислений и могут выполнять только пользовательские задачи.

# **12. Симметричные мультипроцессоры**

Симметричные мультипроцессоры. Памяти находится всего одна копия ОС. Все ЦП равнозначны. При системном вызове на каком-то ЦП происходит прерывание и переход в режим ядра для обработки вызова на этом же ЦП. Для обеспечения взаимоисключений необходимо выделить ряд критических областей в самой ОС и связать их мьютексами.

# **13. В распределенной ОС**

Распределенная ОС существует как единая операционная система в масштабах вычислительной системы. Каждый компьютер сети, работающей под управлением распределенной ОС, выполняет часть функций этой глобальной ОС. Распределенная ОС объединяет все компьютеры сети в том смысле, что они работают в тесной кооперации друг с другом для эффективного использования всех ресурсов компьютерной сети. При работе с распределенной ОС, пользователь не знает на какой машине выполняется его задание или хранятся те или иные данные.

# **14. Серверная ОС**

Серверные операционные системы – специально разработанные или доработанные операционки, служащие для работы веб технологий (ну и не только веб, но в основном). На платформе серверных ОС работает специальное программное обеспечение (программная среда), позволяющее действовать скриптам (оболочкам) сайтов и баз данных, а так же иметь доступ к их файлам через свой компьютер.

# **15. Сеть с равноправными узлами называется**

Ответ: Одноранговая сеть

# **16. Сеть, управляемая одним или несколькими серверами, называется**

Ответ: Сеть с выделенными серверами

# **17. Сети, сочетающие в себе качества одноранговых и сетей на основе сервера, называются**

Ответ: Гибридная сеть

# **18. Ethernet это**

Ethernet — это протокол, который позволяет сетевым устройствам отправлять и получать данные другим устройствам в той же сети. Он обычно используется в сетях, где местоположение является фактором.

# **19. Сетевая служба это**

Ответ: Это совокупность серверной и клиентской частей ОС, предоставляющих доступ к конкретному ресурсу

# **20. К сетевым средствам относятся**

Ответ: 1)серверная часть ОС;2) клиентская часть ОС; 3) транспортные средства ОС

# **21. Гипервизор первого типа это**

Ответ: Гипервизор типа 1 похож на ОС и представляет собой единственную программу запущенную на данном компьютере в привилегированном режиме.

# **22. Гипервизор второго типа это**

Ответ: Гипервизор 2 типа – это программа, которая при управлении ресурсами опирается на существующую ОС и похожа на обычный процесс.

# **23. …aaS означает**

Ответ: as a Service

# **24. IaaS модель**

Ответ: Для этой модели обслуживания провайдер предоставляет клиенту широкие возможности по аппаратному конфигурированию узла. Всё необходимо ПО клиент устанавливает сам.

# **25. PaaS в сравнении с IaaS**

Ответ: Для PaaS модели провайдер предоставляет среду, включающую конкретную ОС, БД, веб-сервер и т.д.

# **26. Основной недостаток SaaS**

Ответ: Предоставляется к конкретному ПО.

Модель SaaS имеет и некоторые слабые места:

* Работоспособность и скорость работы SaaS-сервиса напрямую зависит от состояния и доступности интернет-каналов. Нет широкополосного интернета — нет доступа к услуге.
* Поскольку SaaS, как правило, предоставляется по интернет-каналам общего пользования, скорость доступа к данным и приложениям будет относительно низкой, особенно в сравнении с локальными системами.
* Тот факт, что данные предаются по общедоступным каналам связи, требует повышенного внимания к вопросам информационной безопасности.
* Нет локальной базы данных, а все запросы к БД обрабатываются на стороне оператора, не всегда достаточно быстро.
* Функциональность SaaS-сервиса ограничена возможностями предоставленными оператором. Добавить или убрать что-то принципиальное, как правило, невозможно.

# **27. Такой модели развертывания нет**

Ответ: Есть модели:

* IaaS - «инфраструктура как услуга». ...
* PaaS - «платформа как услуга». ...
* SaaS - «программное обеспечение как услуга». ...
* DRaaS - «аварийное восстановление как услуга».

# **28. Частные облака**

Частное облако — инфраструктура, предназначенная для использования одной организацией, включающей несколько потребителей (например, подразделений одной организации), возможно также клиентами и подрядчиками данной организации.

# **29. Google Cloud или Microsoft Azure предоставляют**

Google Cloud Platform предоставляет такие услуги, как инфраструктура как услуга, платформа как услуга, и бессерверные вычисления.

Это был первый облачный сервис, представленный компанией.

Microsoft Azure позволяет создавать, внедрять, масштабировать приложения и управлять ими в глобальной сети центров обработки данных

# **30. IaaS — это**

IaaS - «инфраструктура как услуга». ...

# **31. PaaS — это**

PaaS - «платформа как услуга». ...

# **32. SaaS — это**

SaaS - «программное обеспечение как услуга». ...

# **33. К характеристикам облачных технологий не относится**

Ответ: Основные характеристики облачных технологий:

1. Самообслуживание по требованию – пользователи должны получать ресурсы без необходимости человеческого участия;
2. Широкий доступ по сети – все ресурсы должны быть доступны по сети, чтобы ими могли воспользоваться гетерогенные устройства;
3. Объединение ресурсов в пул – компьютерные ресурсы, принадлежащие поставщику, должны быть объединены в пул для возможности использования несколькими пользователями с динамическим выделением и освобождением ресурсов;
4. Быстрая эластичность – незамедлительное масштабирование;
5. Учёт услуг – поставщик ведёт учёт услуг способом, соответствующим их типу.

# **34. Arduino — это**

Arduino — это небольшая плата с собственным процессором и памятью.

# **35. Industry 4.0 характеризуется**

Индустрия 4.0 характеризует текущий тренд развития автоматизации и обмена данными, который включает в себя киберфизические системы, Интернет Вещей и облачные вычисления.

# **36. IoT – это**

Ответ: Интернет вещей (IoT – Internet of Things) – сеть физических объектов, которые имеют программные, электронные, сетевые и сенсорные характеристики, позволяющие им собирать и обмениваться данными без участия человека.

# **37. RFID-метка – это**

RFID (Radio Frequency IDentification, радиочастотная идентификация) — способ автоматической идентификации объектов, в котором посредством радиосигналов считываются или записываются данные, хранящиеся в так называемых транспондерах, или RFID-метках.

# **38. К предпосылкам развития Интернета вещей можно отнести**

Ответ: Технологии, повлиявшие на развитие интернет вещей:

1. Развитие процессоров
2. Развитие сетевых технологий
3. Появление различных сенсоров
4. Развитие технологий обработки данных (в частности, машинное обучение ИИ)

# **39. К примерам проектов IoT можно отнести**

* Умный дом
* IoT-устройства в агрокультуре
* Интернет вещей в промышленности
* IoT в ритейле
* Интернет вещей в здравоохранении
* Умные машины
* Носимые устройства
* Умный город
* IoT и каналы поставок

# **40. К типичным примерам применения технологии IoT относится**

Ответ: Интернет вещей создает умные устройства, которые улучшают аспекты повседневной жизни. (Смотри 39 Вопрос)

# **41. Ключевая особенность технологии IoT – это**

Ключевая особенность — в сети будут участвовать не только компьютеры, смартфоны и иные достаточно мощные вычислительные устройства, а практически любые вещи.

# **42. Основной недостаток технологии IoT**

Ответ: Недостатки IoT:

1. Безопасность
2. Сложность – трудоёмкость в развертывании и поддержке
3. Недостаток гибкости

# **43. Технология IIoT – это**

Industrial Internet of Things, IIoT) — это система объединенных компьютерных сетей и подключенных к ним промышленных (производственных) объектов со встроенными датчиками и программным обеспечением для сбора и обмена данными, с возможностью удаленного контроля и управления в автоматизированном режиме, без участия человека.

# **44. К основным компонентам технологии IoT не относится**

Ответ: Основные компоненты систем IoT:

1. Сенсоры и устройства
2. Линии связи
3. Обработка данных
4. Пользовательский интерфейс

# **45. Arduino Uno – это**

Arduino Uno — флагманская платформа для разработки на языке программирования С++. Uno выполнена на микроконтроллере ATmega328P с тактовой частотой 16 МГц. На плате предусмотрены 20 портов входа-выхода для подключения внешних устройств, например плат расширения или датчиков.

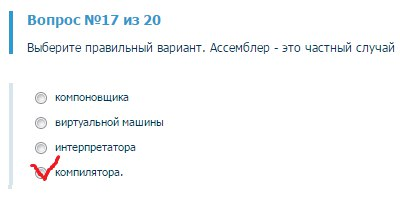
# **46. Устройства IoT получают данные от внешней среды с помощью**

С помощью датчиков устройства Интернета вещей получают данные из своей среды.

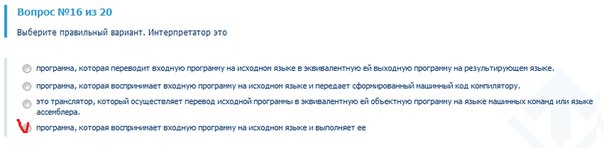
# **47. Ключевая особенность микроконтроллера ESP8266 это**

ESP8266 — микроконтроллер китайского производителя Espressif Systems с интерфейсом Wi-Fi. Помимо Wi-Fi, микроконтроллер отличается отсутствием[1] флеш-памяти в SoC, программы пользователя исполняются из внешней флеш-памяти с интерфейсом SPI.

# **48. Ассемблер - это частный случай**



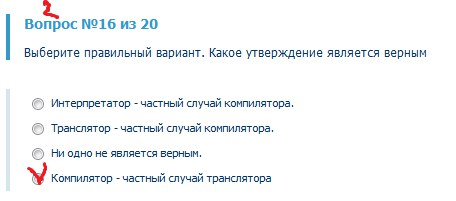
# **49. Интерпретатор это**



# **50. Что такое препроцессор**



# **51. Какое утверждение является верным (вопрос об отношении между компилятором, транслятором и интерпретатором)**

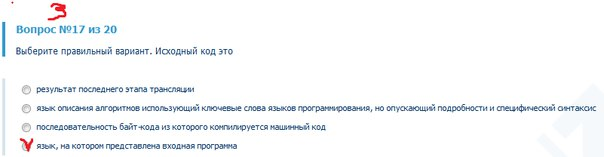


# **52. Какое утверждение является неверным (вопрос об отношении между компилятором, транслятором и интерпретатором)**

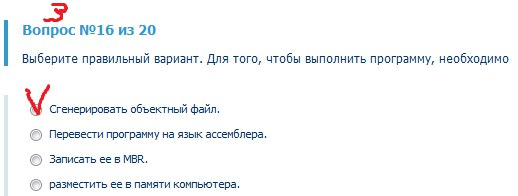
# **53. На последнем этапе трансляции исходный код превращается в**

Ответ: Выполняемый код

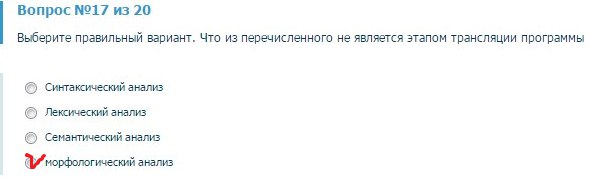
# **54. Исходный код это**



# **55. Для того, чтобы выполнить программу, необходимо**

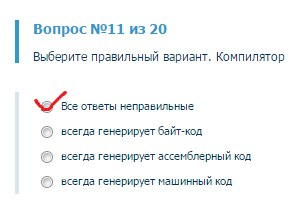


# **56. Что из перечисленного не является этапом трансляции программы**



# **57. Компиляция это**





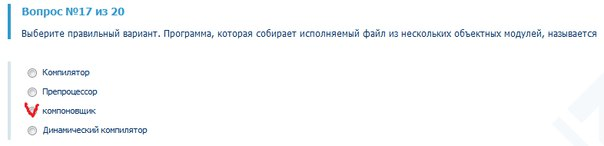
# **58. Программа, позволяющая управлять процессом выполнения программы и обнаруживать в ней ошибки**



# **59. gcc это**



# **60. Программа, которая собирает исполняемый файл из нескольких объектных модулей, называется**



# **61. К общим синтаксическим критериям не относится**

Ответ: На этом этапе лексемы используются для идентификации более крупных программных структур. Сначала синтаксический анализатор идентифицирует последовательность лексем, формирующих синтаксическую единицу. После выполнения такого рода идентификации запускается семантический анализатор.

Теоретической основой для синтаксических анализаторов является теория формальной грамматики.

# **62. К функциям семантического анализатора не относится**

Ответ: Основные функции семантического анализатора:

1. Поддержка таблицы символов
2. Включение неявной информации
3. Обнаружение ошибок
4. Макрообработка и операции выполняемые в процессе компиляции

# **63. python это**

Python — высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ.

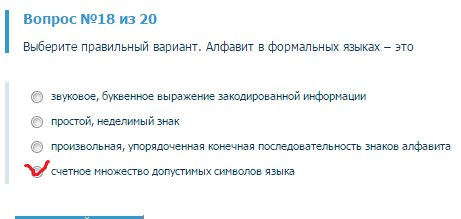
# **64. Необязательным этапом трансляции программы является**

Ответ: Оптимизация – является необязательным этапом трансляции, на котором производятся изменения программы в промежуточном коде с целью повышения эффективности её выполнения.

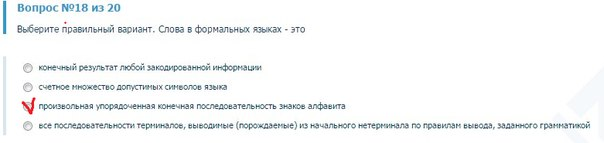
# **65. Символ (буква) в формальных языках – это**



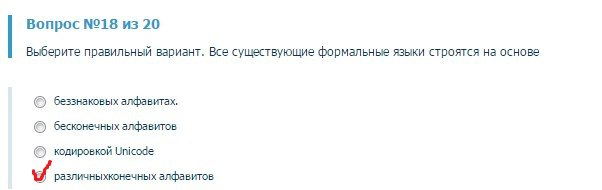
# **66. Алфавит в формальных языках – это**



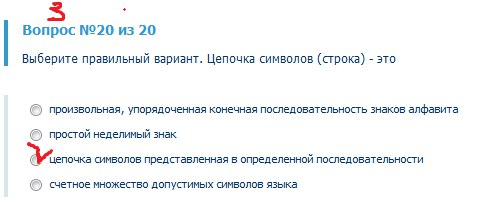
# **67. Слова в формальных языках – это**



# **68. Все существующие формальные языки строятся на основе**



# **69. Цепочка символов (строка) – это**

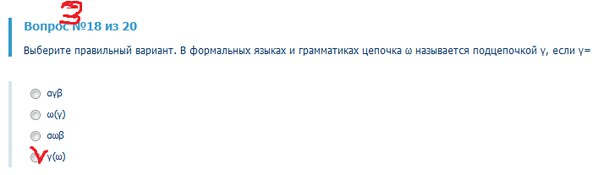


# **70. Длиной цепочки называется**

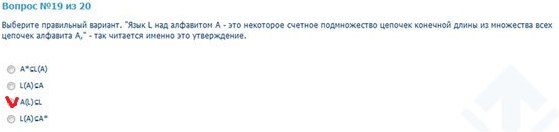
Ответ: Длина цепочки – число позиций для символов в цепочке.

Пример |0 1| = 2, где 2 – длина цепочки

# **71. В формальных языках и грамматиках цепочка ω называется подцепочкой γ, если γ=**



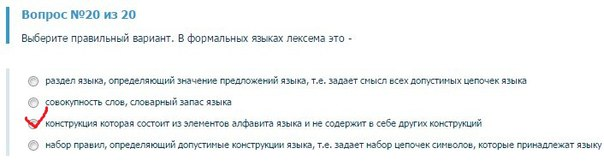
# **72. Язык L над алфавитом А: L(A) это некоторое счетное подмножество цепочек конечной длины из множества всех цепочек алфавита А: (in — является подмножеством)**

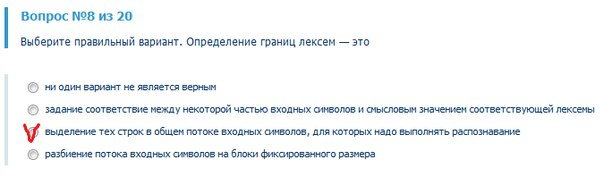


# **73. В формальных языках лексика это –**

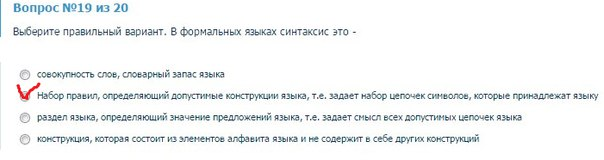


# **74. В формальных языках лексема это –**





# **75. В формальных языках синтаксис это –**



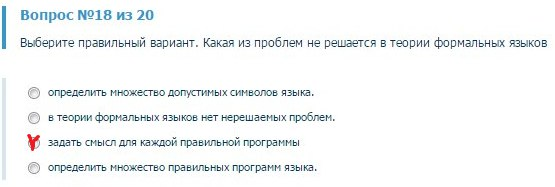
# **76. В формальных языках семантика это –**



# **77. Для задания ЯП необходимо**



# **78. Какая из проблем не решается в теории формальных языков**



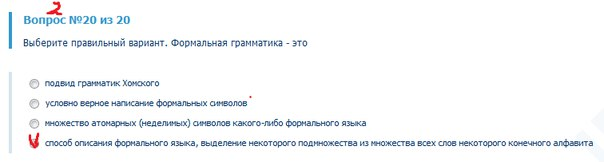
# **79. Что можно описать в виде строгого набора правил?**

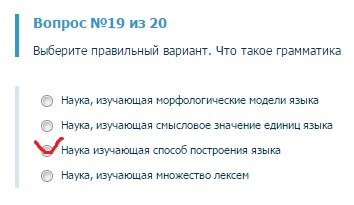


# **80. Язык L над алфавитом А – это**



# **81. Формальная грамматика – это**





# **82. Форма Бэкуса-Наура используется для**





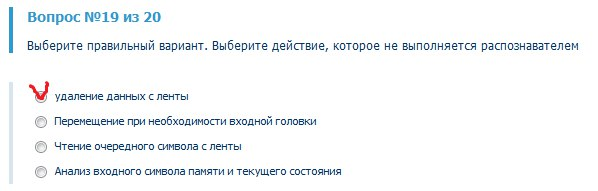
# **83. Распознаватель – это**



# **84. По типу считывающего устройства распознаватели разделяются на**



# **85. Выберите действие, которое не выполняется распознавателем**



# **86. К структурным компонентам распознавателя не относится**

Ответ: К структурным компонентам распознавателя относится:

1. Состояние управляющего устройства
2. Содержимое входной ленты и положение входной головки
3. Содержимое вспомогательной памяти

# **87. Управляющее устройство называется детерминированным, если**

Ответ: если для каждой конфигурации распознавателя существует не более одного следующего шага, иначе распознаватель называется недетерминированным.

# **88. Конфигурация распознавателя называется начальной, если**

Ответ: если управляющее устройство находится в заданном начальном состоянии, входная головка обозревает самый левый символ на входной ленте, память находится в начальном состоянии.

# **89. Формальная грамматика определяется с помощью**

Формальная грамматика или просто грамматика в теории формальных языков — способ описания формального языка, то есть выделения некоторого подмножества из множества всех слов некоторого конечного алфавита. Различают порождающие и распознающие (или аналитические) грамматики — первые задают правила, с помощью которых можно построить любое слово языка, а вторые позволяют по данному слову определить, входит ли оно в язык или нет.

# **90. К метасимволам языка РБНФ не относится**

Ответ: Относятся: цифры, буквы, идентификаторы, тип переменных, ::= , <>, [], {},{| |},(), кавычки.

# **91. Конфигурация распознавателя называется конечной, если**

Ответ: если управляющее устройство находится в одном из заключительных состояний, а входная головка сошла с правого конца входной ленты. Содержимое вспомогательной памяти имеет некоторое удовлетворяющее определенному условию состояние.

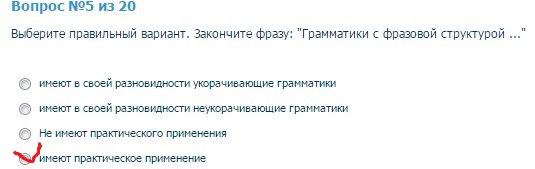
# **92. Язык, определяемый распознавателем, это**

Язык, определяемый распознавателем, — это множество всех цепочек, которые допускает распознаватель.

# **93. Грамматики по Хомскому типа 0 это –**



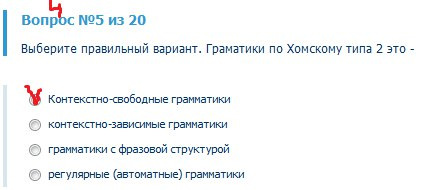
# **94. Закончите фразу: "Грамматики с фразовой структурой ..."**



# **95. Грамматики по Хомскому типа 1 это –**

Тип 1 — контекстно-зависимые грамматики

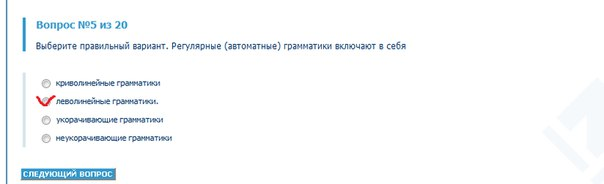
# **96. Грамматики по Хомскому типа 2 это –**



# **97. Грамматики по Хомскому типа 3 это –**

Тип 3 — регулярные грамматики

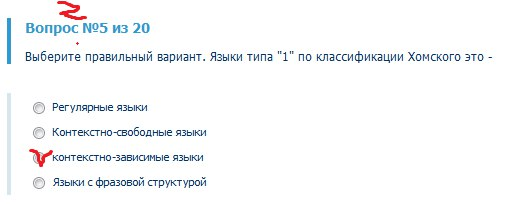
# **98. Регулярные грамматики включают в себя**



# **99. Языки типа "0" по классификации Хомского это –**

Тип 0 — неограниченные языки

# **100. Языки типа "1" по классификации Хомского это –**



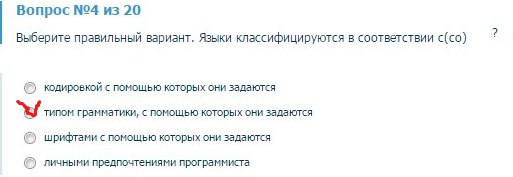
# **101. Языки типа "2" по классификации Хомского это –**

Тип 2 — контекстно-свободные языки

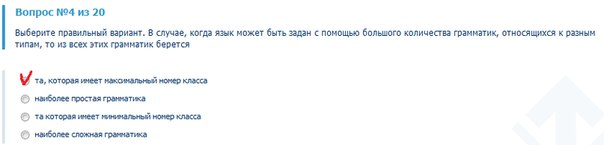
# **102. Языки типа "3" по классификации Хомского это –**



# **103. Языки классифицируются в соответствии с(со)**



# **104. В случае, когда язык может быть задан с помощью большого количества грамматик, относящихся к разным типам, то из всех этих грамматик берется**



# **105. Определите тип языка, заданного правилами вывода: 1) S -> aaCFD; 2) AD -> D; 3) F -> AFB| AB; 4) Cb -> bC; 5) AB -> bBA; 6) CB -> C; 7) Ab -> bA; 8) bCD -> e, где e - цепочка нулевой длины**

Ответ: 0-ой тип языка

# **106. Определите тип языка, заданного правилами вывода: 1) S -> At | Bt; 2) A -> a | Ba; 3) B -> b | Bb | Ab.**

Ответ: 3-ий тип языка

# **107. Определите тип языка, заданного правилами вывода: 1) S -> APA; 2) P -> "+" | -; 3) A -> a | b.**

Ответ: 2-ой тип языка

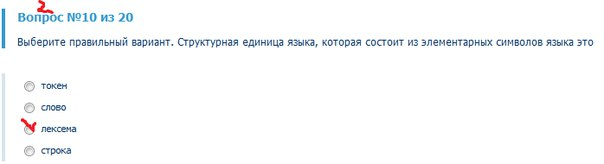
# **108. Определите тип языка, заданного правилами вывода: 1) S -> 1B; 2) B -> B0 | 1.**

Ответ: 2-ой тип языка

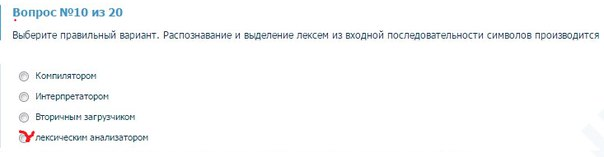
# **109. Определите тип языка, заданного правилами вывода: 1) S -> aSBC | abC; 2) CB -> BC; 3) bB -> bb; 4) bC -> bc; 5) cC -> cc.**

Ответ: 1-ый тип языка

# **110. Структурная единица языка, которая состоит из элементарных символов языка это**



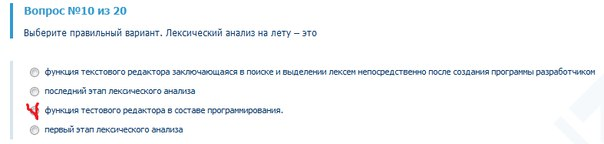
# **111. Распознавание и выделение лексем из входной последовательности символов производится**



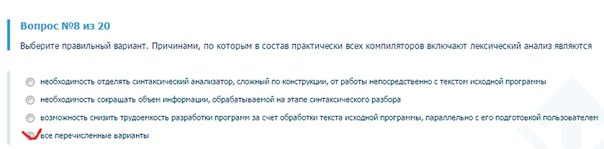
# **112. Лексический анализ**



# **113. Лексический анализ на лету – это**



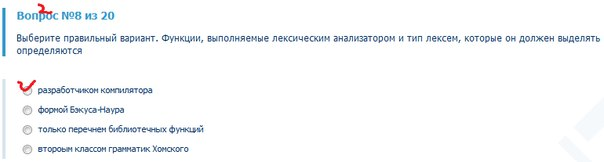
# **114. Причинами, по которым в состав практически всех компиляторов включают лексический анализ являются**



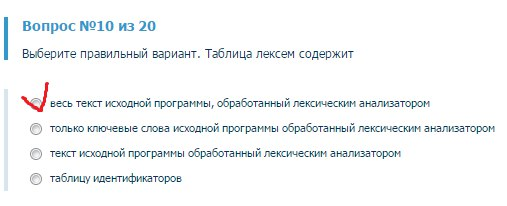
# **115. В ЯП Pascal else является**

Условный оператор в Паскале — if — служит для организации хода задачи таким образом, при котором изменяется последовательность выполнения операторов в зависимости от какого-либо логического условия

# **116. Функции, выполняемые лексическим анализатором и тип лексем, которые он должен выделять определяются**



# **117. Таблица (список) лексем содержит**



# **118. Если изменится кодировка исходной программы, то это повлияет**

Ответ: на оптимизацию компилятора

# **119. К типам лексем ЯП не относится**

К типам лексем ЯП относятся:

a) srting — тип данных «строка»,

b) number — тип данных «число»,

c) boolean — логический тип данных,

d) undefined — тип данных, значения которых не определены,

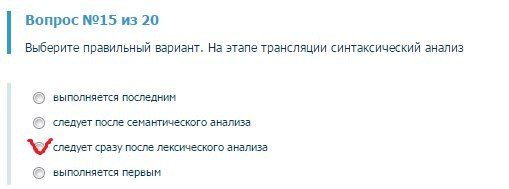
e) null — тип данных с «пустыми» значениями,

f) object — тип данных, хранящих свойства и методы.

# **120. Теоретической моделью, используемой при построении лексического анализатора, является**

Ответ: конечный автомат

# **121. На этапе трансляции синтаксический анализ**

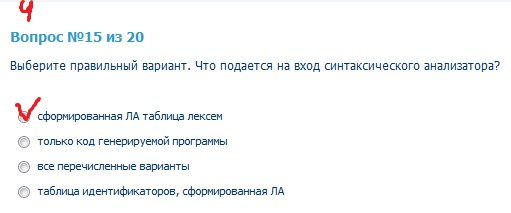


# **122. Синтаксический анализатор – это**

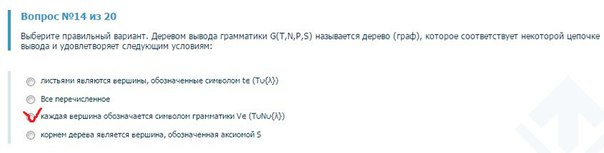




# **123. Что подается на вход синтаксического анализатора?**



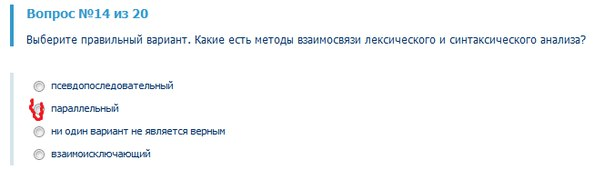
# **124. Деревом вывода грамматики G(T,N,P,S) называется дерево (граф), которое соответствует некоторой цепочке вывода и удовлетворяет следующим условиям:**



# **125. МПА при построении синтаксического анализатора – это**



# **126. Одним из самых простых методов синтаксического анализа является**



# **127. Характерной особенностью автомата с магазинной памятью является**

Магазинная память определяется свойством «первым введен — последним выведен». При записи символа в магазин, его содержимое сдвигается на одну ячейку «вниз», а на освободившееся место записывается требуемый символ. В этом случае говорят, что символ «вталкивается» в магазин.

# **128. Метод рекурсивного спуска применим к**

Метод рекурсивного спуска применим к достаточно узкому подклассу КС-грамматик. Известны более широкие подклассы КС-грамматик, для которых существуют эффективные анализаторы, обладающие тем же свойством, что и анализатор, написанный методом рекурсивного спуска, - входная цепочка считывается один раз слева направо и процесс разбора полностью детерминирован, в результате на обработку цепочки длины n расходуется время cn.

# **129. Синтаксис языков программирования описывают с помощью**

Cинтаксиса языка строится с использованием лексем – минимальных значимых единиц-элементов текста. Лексемы, которые являются названиями конструкций языка, называются нетерминальными символами или просто нетерминалами (например, <оператор>, <число>, <условный оператор> и т. д.). Число нетерминалов равно числу правил языка.

# **130. Детерминированный МПА**

МПА называется детерминированным, если из каждой его конфигурации возможно не более одного перехода в другую конфигурацию. Формально для ДМПА функция переходов δ может иметь один из трех видов:

1.δ(q,a,z) содержит 1 элемент: δ(q,a,z)={(q’,γ)}, γ Z\*, δ(q,λ,z)= ;

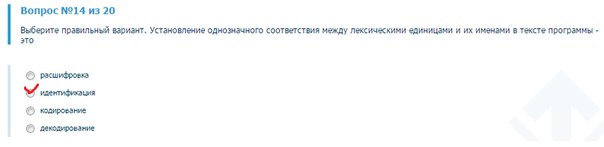
2.δ(q,a,z)= , δ(q,λ,z) содержит 1 элемент δ(q,λ,z)={(q’,γ)}, γ Z\*;

3.δ(q,a,z)= , δ(q,λ,z)= ;

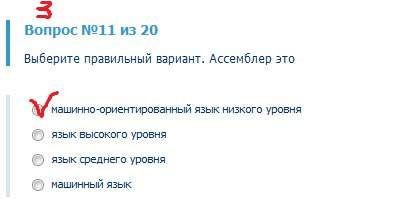
# **131. Недетерминированный МПА**

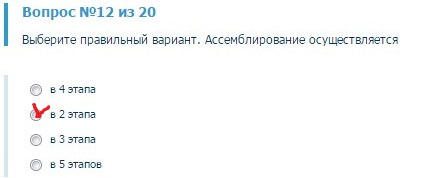
Недетерминированный МПА - устройство, реализующее перебор с возвратами, то есть неэффективный механизм распознавания языка.

# **132. Установление однозначного соответствия между лексическими единицами и их именами в тексте программы – это**

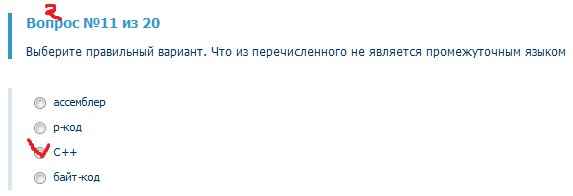


# **133. Ассемблер – это**





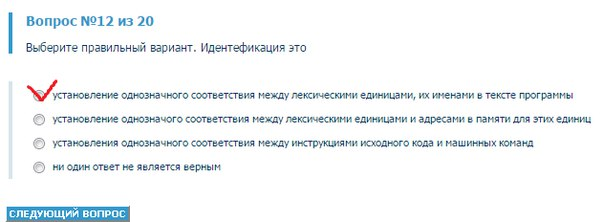
# **134. Что из перечисленного не является промежуточным языком**



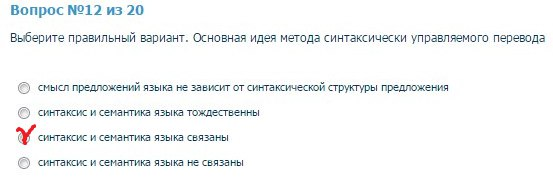
# **135. Генерация кода**



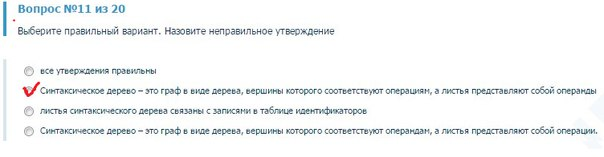
# **136. Идентификация это**



# **137. Основная идея метода синтаксически управляемого перевода**



# **138. Назовите неправильное утверждение**



# **139. Дерево операций**



# **140. Для формирования внутреннего представления программы может применяться**

Генерация внутреннего представления программы будет проходить во время синтаксического анализа параллельно с контролем контекстных условий, поэтому для генерации можно использовать информацию, "собранную" синтаксическим и семантическим анализаторами; например, при генерации ПОЛИЗа выражений можно воспользоваться содержимым стека, с которым работает семантический анализатор.

# **141. ПОЛИЗ — это**

ПОЛИЗ - польская инверсная запись

# **142. Преимущество триад при промежуточном представлении программы перед тетрадами в том, что**

Тетрады требуют больше памяти для своего представления, чем триады, они также не отражают явно взаимосвязь операций между собой. Кроме того, есть нюансы с преобразованием тетрад в машинный код, так как они плохо отображаются в команды ассемблера и машинные коды, поскольку в наборах команд большинства современных компьютеров редко встречаются операции с трек операндами.

# **143. Выберите ПОЛИЗ для выражения (2+5) \* (5-8) / 3**

Ответ: 2 5 + 5 8 - \* 3 /

# **144. Выберите ПОЛИЗ для выражения (4-2) \* 4 / 5**

Ответ: 4 2 - 4 \* 5 /

# **145. Промежуточный код**

Промежуточным кодом в общем называется некоторое внутреннее представление, используемое компилятором при переходе от начальной стадии компиляции к заключительной. Следует отметить, что в общем случае компилятор может иметь несколько промежуточных представлений различного уровня абстракции.

# **146. К достоинствам триадного представления промежуточного кода не относится**

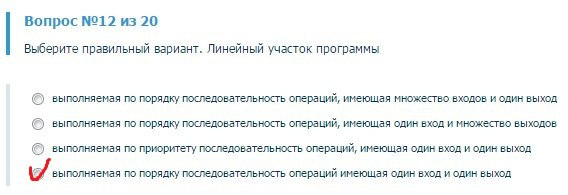
В триадном представлении легко исключить лишние команды.

Прежде всего разделим команды на те, что производят действия и не производят. Очевидно, что триады команд второй группы, на которые нет ссылок в других триадах можно исключить. Затем повторять проверку до тех пор, пока не будут исчерпаны возможности исключения триад.

# **147. К недостаткам тетрадного представления промежуточного кода относится**

Существуют и иные приемы оптимизации на уровне триад, а следовательно и на уровне тетрад, например выявление практически недостижимых команд в силу расположения меток и инструкции останова, замены нескольких триад одной, если это возможно.

# **148. Линейный участок программы**

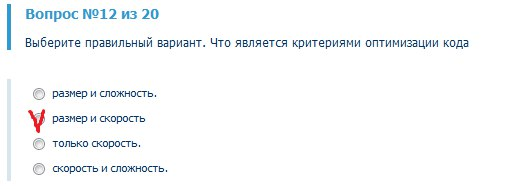


# **149. Для линейных участков не могут выполняться следующие виды оптимизации**

Ответ: Для линейных участков могут выполняться следующие виды оптимизации:

1. Удаление бесполезных присваиваний
2. Исключение избыточных вычислений лишних операций
3. Свертка операций объектного кода
4. Перестановка операций
5. Арифметические преобразования

# **150. Что является критериями оптимизации кода**



# **151. Оптимизация это**

Ответ: Оптимизация – это обработка, связанная с переупорядочиванием и изменением операций в компилируемой программе с целью получения более эффективной результирующей объектной программы.

# **152. К приемам оптимизации линейных участков программы не относится**

Ответ: Для линейных участков могут выполняться следующие виды оптимизации:

1. Удаление бесполезных присваиваний
2. Исключение избыточных вычислений лишних операций
3. Свертка операций объектного кода
4. Перестановка операций
5. Арифметические преобразования

# **153. К приемам оптимизации циклов не относится**

Для оптимизации циклов используются следующие методы:

* вынесение инвариантных вычислений из циклов;
* замена операций с индуктивными переменными;
* слияние и развертывание циклов.

# **154. К приемам оптимизации не относится**

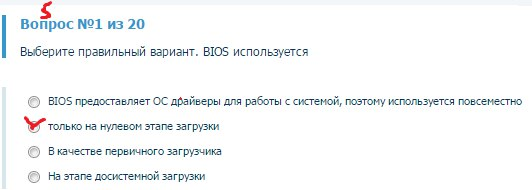
Ответ: Различают два вида оптимизации:

1. Преобразования исходной программы
2. Преобразование результирующей объектной программы

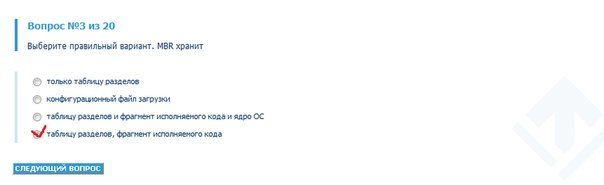
# **155. Особенностью оптимизации как этапа трансляции является**

Ответ: то, что по сути это компиляция

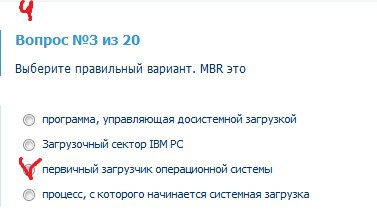
# **156. BIOS используется**



# **157. MBR хранит**



# **158. MBR это**



# **159. Встроенное программное обеспечение**

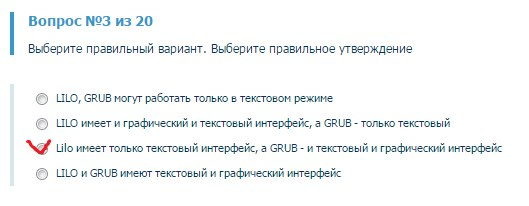


# **160. Вторичный загрузчик**

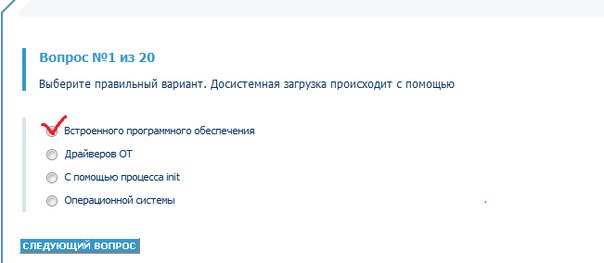


# **161. Выберите истинное утверждение**





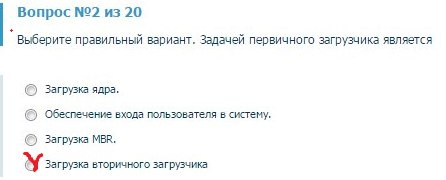
# **162. Досистемная загрузка происходит с помощью**



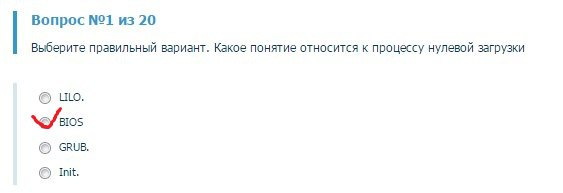
# **163. Задачей вторичного загрузчика является**



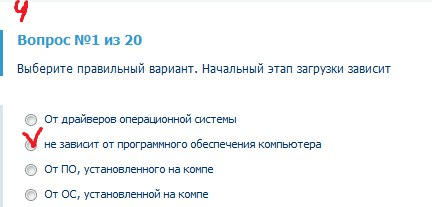
# **164. Задачей первичного загрузчика является**



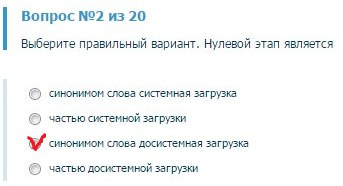
# **165. Какое понятие относится к процессу нулевой загрузки**



# **166. Начальный этап загрузки зависит**



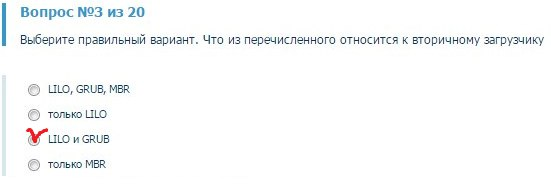
# **167. Нулевой этап загрузки является**



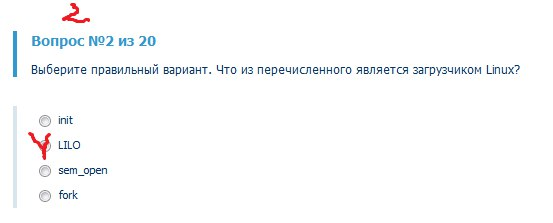
# **168. Работа вторичного загрузчика**



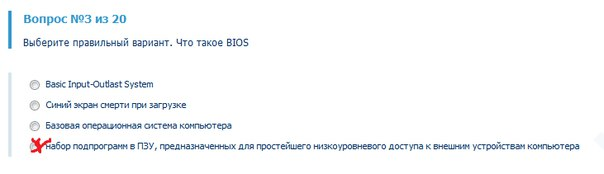
# **169. Что из перечисленного относится к вторичному загрузчику**



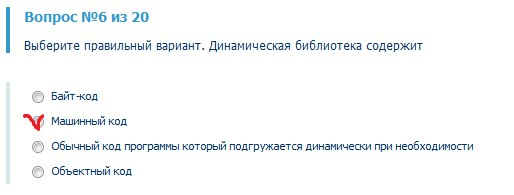
# **170. Что из перечисленного является загрузчиком Linux?**



# **171. Что такое BIOS**



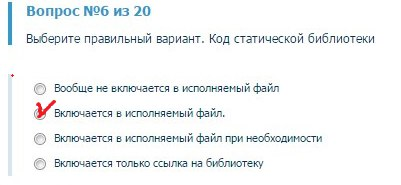
# **172. Динамическая библиотека содержит**



# **173. Изменение динамической библиотеки, влияет на программы которые ее используют?**



# **174. Код статической библиотеки**



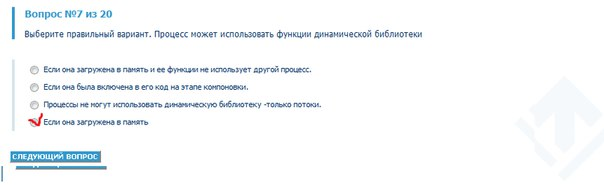
# **175. Отличительный признак динамической библиотеки**



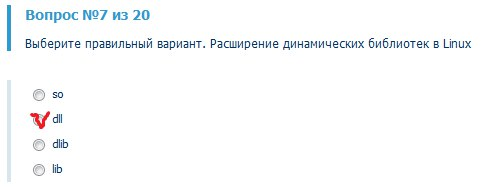
# **176. При использовании статической библиотеки**



# **177. Процесс может использовать функции динамической библиотеки**



# **178. Расширение динамических библиотек в Linux**



# **179. Расширение статических библиотек в Linux**



# **180. Для указания импортируемых из DLL функций применяют конструкцию**

Ответ: \_\_declspec(dllimport)

# **181. Для указания экспортируемых из DLL функций применяют конструкцию**

Ответ: \_\_declspec(dllexport)

# **182. При загрузке DLL c неявной компоновкой**

Чтобы вызвать функцию из DLL, ее необходимо объявить в вызывающем коде - либо как external, либо предварить ключевым словом \_\_declspec(dllimport).

# **183. При загрузке DLL c явной компоновкой**

Явную загрузку динамических библиотек осуществляет функция HINSTANCE LoadLibrary(LPCTSTR lpLibFileName) или ее расширенный аналог HINSTANCE LoadLibraryEx(LPCTSTR lpLibFileName, HANDLE hFile, DWORD dwFlags).

# **184. Расширение динамических библиотек в Windows**

Динамические библиотеки обычно они имеют расширение \*.dll в Windows.

# **185. Расширение статических библиотек в Windows**

В Windows статические библиотеки обычно имеют расширение \*.lib.

# **186. Функция входа в DLL — это функция**

Функция DllMain - дополнительная точка входа в динамически-подключаемую библиотеку (DLL). Если функция используется, то она вызывается системой тогда, когда процессы и потоки инициализированы и завершили работу или при вызове функции LoadLibrary и FreeLibrary. DllMain - имя - заместитель для определяемого библиотекой имени функции.

# **187. К достоинствам использования динамических библиотек относят**

Ответ: Преимущества использования динамических библиотек:

1. Простота исправления динамических библиотек
2. Экономия памяти
3. Экономия места на диске
4. Централизованное представление общих данных
5. Простое расширение функциональности приложения
6. Возможность использования разных языков программирования
7. Ускорение процесса сборки
8. Разработка в большой команде

# **188. К недостаткам использования динамических библиотек относят**

Ответ: Недостатки:

1. Неудобство развертывания
2. Потеря производительности
3. Проблемы с зависимостями
4. Неудобство при сборке

# **189. Сравните файлы ls и ln, находящиеся в каталоге /bin, используя абсолютные имена файлов**

Ответ: cmp -b -l /bin/ln /bin/ls (еще варик diff <(xxd /bin/ln) <(xxd /bin/ls))

# **190. Вы находитесь в каталоге ~/. Просмотрите первые 10 строк файла /etc/group, используя относительное имя файла**

Ответ: head -n10 ../../etc/group

# **191. Вы находитесь в каталоге /var/opt. Просмотрите первые 10 строк файла /home/user/mytxt, используя относительное имя файла**

Ответ: head -n10 ../../home/user/mytxt

# **192. Вы находитесь в каталоге /opt/pycharm. Просмотрите первые 10 строк файла ~/mytxt, используя абсолютное имя файла**

Ответ: head -n10 /home/user/mytxt

# **193. Вы находитесь в каталоге ~. Где вы окажетесь после применения команды cd ../../../../../../../..**

Ответ: / - корень системы

# **194. Вы находитесь в каталоге ~. Где вы окажетесь после применения команды cd ../././././././vasia/./.././brother**

Ответ: /home/brother

# **195. Вы находитесь в каталоге ~. Где вы окажетесь после применения команды cd 1/2/3/././4/././..**

Ответ: ~/1/2/3

# **196. Вы находитесь в каталоге /usr/local/bin. Где вы окажетесь после применения команды cd /../../../../etc/././..**

Ответ: - корень системы

# **197. Вы находитесь в каталоге /bin. Где вы окажетесь после применения команды cd /../var/./././**

Ответ: /usr/local/var

# **198. Поменяйте владельца и группу файла ~/mytxt на root**

Ответ: sudo chown root ~/mytxt; sudo chown .root ~/mytxt

# **199. Установите, используя символы прав доступа следующие права на файл ~/mytxt: для себя все права, для группы и остальных — никаких**

Ответ: sudo chmod u+rwx ~/mytxt; sudo chmod go-rwx ~/mytxt

# **200. Установите при помощи восьмеричных масок следующие права на файл ~/mytxt: для себя все права, для группы и остальных — никаких**

Ответ: sudo chmod 700 ~/mytxt

# **201. Установите при помощи восьмеричных масок следующие права на файл ~/mytxt: для себя исполнение и запись, для группы никаких, для остальных чтение**

Ответ: sudo chmod 304 ~/mytxt

# **202. Просмотрите руководство по команде more**

Ответ: man more

# **203. Создайте жесткую ссылку для файла /home/user1/mytxt с тем же именем, поместив ее в каталог /home/user1/work**

Ответ: ln -P /home/user1/mytxt /home/user1/work/mytxt

# **204. Создайте символьную ссылку для файла /home/user1/mytxt с тем же именем, поместив ее в каталог /home/user1/work**

Ответ: ln -s /home/user1/mytxt /home/user1/work/mytxt

# **205. Получите подробную информацию о файлах в текущем каталоге**

Ответ: ls -l или ll

# **206. Получите список файлов текущего каталога, отсортированный по размеру**

Ответ: ls –alS

# **207. Выведите на консоль имя самого большого (по размеру) файла текущего каталога**

Ответ: ls -S| head -n2 | tail -n1

# **208. Найдите все текстовые файлы (с расширением txt) текущего каталога, содержащие в названии "pro"**

Ответ: ls | grep'pro\*.\*\.txt$'

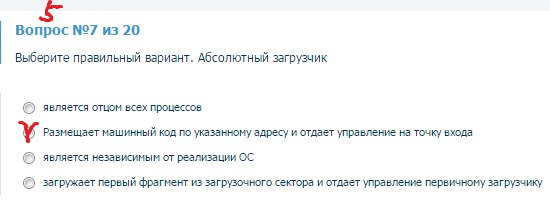
# **209. Выведите список пользователей в алфавитном порядке**

Ответ: sed 's/:.\*//' /etc/passwd | sort

# **210. Получите информацию о текущем пользователе**

Ответ: whoami (в ubuntu выведет просто пользователя, там больше ничего нет по функционалу)

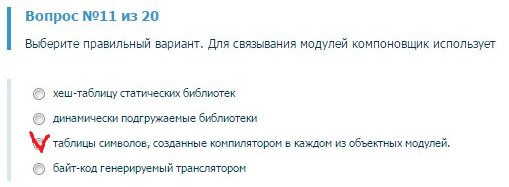
# **211. Абсолютный загрузчик**



# **212. Динамический загрузчик выполняет связывание**

Динамическое связывание относится к связыванию, которое выполняется во время загрузки или выполнения, а не при создании исполняемого файла. В случае динамической компоновки компоновщик при создании exe выполняет минимальную работу. Для того, чтобы динамический компоновщик работал, ему фактически также необходимо загрузить библиотеки, поэтому его также называют загрузчиком компоновки.

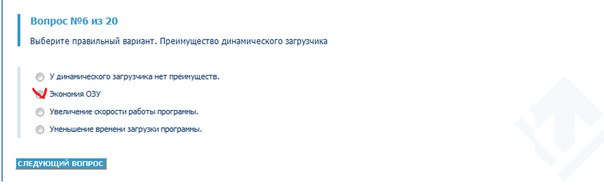
# **213. Для связывания модулей компоновщик использует**



# **214. Недостаток динамического загрузчика**



# **215. Преимущество динамического загрузчика**



# **216. Редактор связей**



# **217. Преимущество редактора связей**



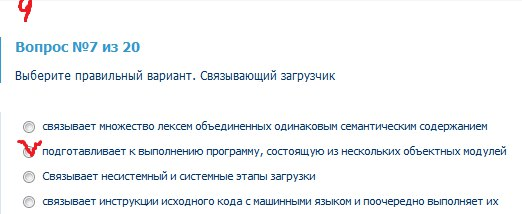
# **218. Недостатки редактора связей**

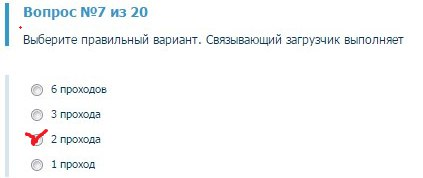


# **219. Редактор связей выполняет связывание**

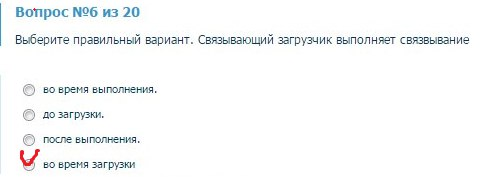


# **220. Связывающий загрузчик**





# **221. Связывающий загрузчик выполняет связывание**



# **222. Что не является преимуществом связывающего загрузчика**

Ответ: Такой загрузчик обладает тем **преимуществом**, что допускает использование большого числа как программных сегментов, так и сегментов данных и дает программисту полную свободу обращения к данным, находящимся в других сегментах, допуская при этом раздельную трансляцию.

# **223. Что является преимуществом связывающего загрузчика**

