Вопросы к экзамену

1. Понятие о данных.
2. Понятия программы, алгоритма.
3. Понятия типа данных, системы типов языка программирования, типизации.
4. Важнейший парадигмы программирования и их отличительные черты.
5. Каким образом в язык программирования с динамической типизацией можно ввести новый тип данных? Приведите примеры.
6. Понятие абстрактного типа данных. Примеры.
7. Функции (процедуры) высшего порядка в языках программирования высокого уровня.
8. Способы организации повторяющихся вычислений в языках программирования высокого уровня.
9. Мемоизация результатов вычислений.
10. Нестрогие и отложенные вычисления. Примеры.
11. Способы реализации языка программирования высокого уровня.
12. Компилятор и интерпретатор: определение, основные функциональные элементы.
13. Лексический анализатор: назначение, входные данные, выходные данные, принцип реализации.
14. Синтаксический анализатор: назначение, входные данные, выходные данные.
15. Формальная грамматика: терминальные символы, нетерминальные символы, правила, аксиома.
16. БНФ и синтаксические диаграммы.
17. $LL(1)$-грамматика: особенности и их использование.
18. Принцип построения лексического анализатора.
19. Принцип построения нисходящего синтаксического анализатора, осуществляющего разбор методом рекурсивного спуска.
20. Основные понятия объектно-ориентированного программирования.
21. Модульное тестирование (юнит-тестирование), разработка через тестирование.

22. Способы разбора и вычисления значения арифметического выражения, записанного в инфиксной нотации, с учетом приоритетов операций и скобок.
23. Основные постулаты языков программирования семейства Lisp.
24. Общая характеристика языка Scheme.
25. Типизация и система типов языка Scheme.
26. Способы определения процедуры в языке Scheme. Формальные и фактические аргументы, применение к аргументам, возвращаемое значение.
27. Простые типы языка Scheme и основные операции над ними.
28. Составные типы языка Scheme и основные операции над ними.
29. Символьный тип в языке Scheme и его применение.
30. Применение процедур (функций) высшего порядка для обработки списков на языке Scheme.
31. Лексические замыкания (на примере) в языке Scheme. Свободные и связанные переменные. Использование лексических замыканий для локальных определений (запись конструкций `let` и `let*` с помощью анонимных процедур).
32. Лексические замыкания (на примере) в языке Scheme. Свободные и связанные переменные. Использование лексического замыкания для определения процедуры со статической переменной.
33. Особенности логических операций в языке программирования Scheme.
34. Гигиенические макросы в языке Scheme.
35. Продолжения в языке Scheme.
36. Ввод-вывод в языке Scheme.
37. Средства для метапрограммирования языка Scheme.
38. Хвостовая рекурсия и ее оптимизация интерпретатором языка Scheme.
39. Основные управляющие конструкции языка Scheme.
40. Ассоциативные списки.
41. Точечные пары и списки в языках семейства Lisp.
42. Командный интерпретатор Bash: общая характеристика языка, пример сценария командной оболочки.

43. Действия программиста для создания сценария («скрипта») на интерпретируемом языке программирования, предназначенного для запуска из командной оболочки UNIX-подобной операционной системы.
44. Стандартные потоки ввода-вывода, аргументы командной строки. Перенаправление ввода-вывода в командной оболочке Bash, использование конвейеров (pipes).
45. Файловая система, путь к файлу и атрибуты файла. Основные команды оболочки для работы с файлами и папками.
46. Общая характеристика языка Javascript, типизация и система типов.
47. Понятие объекта. Создание и использование объектов в языке Javascript.
48. Применение функций высшего порядка для обработки последовательностей на языке Python Javascript.
49. Лексические замыкания (на примере) в языке Javascript. Свободные и связанные переменные.
50. Особенности логических операций в языке программирования Javascript.
51. Обработка исключений в языке Javascript.
52. Определение класса и создание объекта класса на языке Javascript.
53. Средства метапрограммирования языка Javascript.
54. Особенности присваивания, копирования и передачи в функцию объектов в языке Javascript.

Примеры задач

Приведены только примеры задач

1. На языке Javascript напишите возможную реализацию встроенной функции `filter` (варианты: `map` для функции одной переменной, `reduce` и пр.).
2. На языке Scheme напишите функцию `drop`, принимающую список и целое число *n* и возвращающую исходный список без *n* первых элементов, например, так:

   `(drop '(1 2 3 4) 2) ⇒ (3 4)`
3. На языке Javascript напишите 2 варианта функции, вычисляющей среднее арифметическое последовательности чисел. В одном варианте используйте императивные управляющие конструкции, во втором -- встроенные функции высших порядков.