Содержание экзамена: два теоретический вопроса + задача

Задача пишется в moodle, теория пишется на листочках - проверяется написанное: чем больше, тем лучше.

задача - 45-60 минут (в зависимости от сложности)

теория - 30 минут

Все примеры должны быть содержательными и полными для максимального балла за ответ.

Цветом выделены комментарии преподавателя по ответам на вопросы.

Вопросы к экзамену:

1. Указатель на void (для чего используется, особенности, примеры). Стандартные функции обработки областей памяти (memcpy, memmove, memcmp, memset). Если остается время можно написать про совместное использование указателя на void совместно с указателями на функции, особенности.
2. Функции динамического выделения памяти (malloc, calloc, free, отдельно - realloc). Рассказать про выделение 0 байт памяти, приведение типов при использовании функций.
3. Выделение памяти под динамический массив (2 способа). Типичные ошибки при работе с динамической памятью (классификация ошибок, примеры). Подходы к обработке ситуации отсутствия динамической памяти. Как вернуть из функции динамически созданный массив?ё
4. Указатели на функцию (для чего используются, примеры, описание указателя на функцию, вызов функции по указателю, особенности адресной арифметики применительно к указателям на функции – целостные примеры). Функция qsort (примеры использования).
5. Утилита make: назначение, простой сценарий сборки. Для чего используется, как работает утилита. Разновидности утилиты make, что такое сценарий сборки, правила. Пример простого сценария сборки, на нем показать работу утилиты.
6. Утилита make: назначение, переменные, шаблонные правила. Для чего нужна, что подается на вход, алгоритм работы, разновидности. Правила, переменные: примеры использования обычных переменных. Неявные переменные и правила, автоматические переменные и шаблонные правила. Примеры.
7. Утилита make: назначение, условные констру кции, анализ зависимостей. Для чего нужна, что подается на вход, алгоритм работы, разновидности. Сценарии сборки, правила, условные конструкции с примерами. Переменные, зависящие от цели. Подходы к анализу зависимостей (выставление вручную; все файлы .c зависят от всех файлов .h; переложение выяснения зависимостей на компилятор (gcc -M)).
8. Динамические матрицы. Представление в виде одномерного массива и массива указателей на строки. Анализ преимуществ и недостатков. Графическая интерпретация, функции выделения памяти для каждого способа, таблица сравнения способа по критериям (критерии из лекции + любые свои здравые критерии).
9. Динамические матрицы. Представление в виде одномерного массива и массива указателей на строки, расположенные единым блоком (первый объединенный способ). Анализ преимуществ и недостатков. Графическая интерпретация, функции выделения памяти для каждого способа, таблица сравнения способа по критериям (критерии из лекции + любые свои здравые критерии).
10. Динамические матрицы. Представление в виде одномерного массива и массива указателей на строки, причем массив указателей и строки расположены единым блоком (второй объединенный способ). Анализ преимуществ и недостатков. Графическая интерпретация, функции выделения памяти для каждого способа, таблица сравнения способа по критериям (критерии из лекции + любые свои здравые критерии).
11. Динамические матрицы. Представление в виде массива указателей на строки и массива указателей на строки, расположенные единым блоком (первый объединенный способ). Анализ преимуществ и недостатков. Графическая интерпретация, функции выделения памяти для каждого способа, таблица сравнения способа по критериям (критерии из лекции + любые свои здравые критерии).
12. Динамические матрицы. Представление в виде массива указателей на строки и массива указателей на строки, причем массив указателей и строки расположены единым блоком (второй объединенный способ). Анализ преимуществ и недостатков. Графическая интерпретация, функции выделения памяти для каждого способа, таблица сравнения способа по критериям (критерии из лекции + любые свои здравые критерии).
13. Динамические матрицы. Представление в виде массива указателей на строки, расположенные единым блоком (первый объединенный способ), и массива указателей на строки, причем массив указателей и строки расположены единым блоком (второй объединенный способ). Анализ преимуществ и недостатков. Графическая интерпретация, функции выделения памяти для каждого способа, таблица сравнения способа по критериям (критерии из лекции + любые свои здравые критерии).
14. Чтение сложных объявлений. Примеры
15. Строки в динамической памяти. Функции POSIX, расширение GNU, возвращающие такие строки (strdup, strndup, getline, snprintf – знать, что делает каждая, и уметь реализовать). Feature Test Macro – для чего нужны, какие разновидности существуют, как эти макросы читаются с учетом стандарта, указываемого при компиляции.
16. Особенности использования структур с полями указателя. Операции присваивания, глубокое, поверхностное копирование, примеры. В каких случаях лучше использовать каждое из них? Рекурсивное освобождение памяти (сначала освобождение внутренних полей, затем самой структуры).
17. Структура переменного размера. Где на практике встречаются, примеры. Поле flexible array member, его особенности и особенности структур с таким полем. Как обходились до появления FAM, сравнение поля-указателя с полем FAM (преимущества и недостатки).
18. Динамически расширяемый массив. Определение массива, описание типа для массива, выделение памяти относительно крупными блоками, добавление элемента в массив, освобождение, особенности использования массива.
19. Линейный односвязный список. Добавление элемента, удаление элемента. Определение линейного односвязного списка, узла, сравнение с массивом, описание функций. Добавление в начало и конец. Освобождение списка.
20. Линейный односвязный список. Вставка элемента, удаление элемента. Определение линейного односвязного списка, узла, сравнение с массивом, описание функций. Вставка перед и после. Освобождение списка.
21. Линейный односвязный список. Обход. Определение линейного односвязного списка, узла, сравнение с массивом. Универсальная функция обхода списка. Освобождение списка.
22. Двоичное дерево поиска (ДДП). Добавление элемента. Определение ДДП, различия ДДП и обычного дерева. Описание узла, освобождение памяти. Рекурсивная реализация функции и не рекурсивная.
23. ДДП. Поиск. Определение ДДП, различия ДДП и обычного дерева. Описание узла, освобождение памяти. Рекурсивная реализация функции и не рекурсивная.
24. ДДП. Обход. Определение ДДП, различия ДДП и обычного дерева. Описание узла, освобождение памяти. Универсальная функция обхода. Если есть время, можно вспомнить про язык DOT.
25. ДДП. Удаление элемента (требуется полноценная функция, чего не было на лекции). Определение ДДП, различия ДДП и обычного дерева. Описание узла, освобождение памяти.
26. Куча в программе на Си. Алгоритм работы функции malloc. Пример реализации. История происхождения термина «куча», для чего используется, преимущества и недостатки динамической памяти. Какие гарантии относительно выделенного блока памяти даются программисту? Алгоритм работы функции malloc, пример реализации.
27. Куча в программе на Си. Алгоритм работы функции free. Пример реализации. История происхождения термина «куча», для чего используется, преимущества и недостатки динамической памяти. Какие гарантии относительно выделенного блока памяти даются программисту? Алгоритм работы функции free. Что с точки зрения функции означает освобождение области памяти? Пример реализации функции free. Что такое дефрагментация? (достаточно определения)
28. Куча в программе на Си. Проблема выравнивания выделенной области памяти. Пример реализации. История происхождения термина «куча», для чего используется, преимущества и недостатки динамической памяти. Какие гарантии относительно выделенного блока памяти даются программисту? Проблема выравнивания, ее решение.
29. Массив переменной длины, функция alloca.
30. Функции с переменным числом параметров. Ответ можно построить по примеру лекции: идея реализации таких функций, почему так делать нельзя, стандартный способ (stdarg), макросы stdarg, для чего нужны, как пользоваться. Как написать функцию, аналогичную printf, с помощью стандартной библиотеки (на примере журналирования)?
31. Препроцессор. Общие понятия. Директива include. Простые макросы. Предопределенные макросы. Что такое препроцессор, в какой момент он обрабатывает программу, какие функции выполняет? Управляется директивами: на какие группы делятся, общие правила при их описании. Include: чем использование двойных кавычек отличается от использования угловых скобок. Простой макрос – обработка препроцессором, примеры, для чего используются. Какие известны предопределенные макросы, для чего используются?
32. Препроцессор. Макросы с параметрами. Что такое препроцессор, в какой момент он обрабатывает программу, какие функции выполняет? Управляется директивами: на какие группы делятся, общие правила при их описании. Как макросы с параметрами обрабатываются препроцессором, сравнение макросов с параметрами и функций, макросы с переменным числом параметров, примеры. Зачем использовать круглые скобки внутри тела макроса с параметрами? Подходы к написанию длинных макросов, примеры.
33. Препроцессор. Общие понятия. Директивы условной компиляции, директивы error и pragma. Что такое препроцессор, в какой момент он обрабатывает программу, какие функции выполняет? Управляется директивами: на какие группы делятся, общие правила при их описании. Для чего используются директивы условной компиляции, какие бывают (сравнить директивы группы if и группы ifdef). #error, #pragma.
34. Препроцессор. Общие понятия. Операции # и ##. Что такое препроцессор, в какой момент он обрабатывает программу, какие функции выполняет? Управляется директивами: на какие группы делятся, общие правила при их описании. Операции # и ##, примеры использования, особенности работы препроцессора при раскрытии макросов.
35. Встраиваемые функции. Ключевое слово inline. Для чего появилось. Сравнение с макросами, функциями. Способы борьбы с ошибкой undefined reference.
36. Библиотеки. Статические библиотеки. Что такое библиотека? Какие функции выносятся в библиотеку? В какой форме распространяются? Какие виды библиотек существуют, их преимущества. Как собрать статическую библиотеку? Как собрать с ней приложение? Ключи -L -l, особенности именования библиотек.
37. Библиотеки. Динамические библиотеки. Динамическая компоновка. Что такое библиотека? Какие функции выносятся в библиотеку? В какой форме распространяются? Какие виды библиотек существуют, их преимущества. Как собрать динамическую библиотеку, особенности в зависимости от ОС. Оформление кода при использовании библиотеки (в зависимости от ОС). Компоновка vs загрузка. Ключи -L -l, особенности именования библиотек.
38. Библиотеки. Динамические библиотеки. Динамическая загрузка. Что такое библиотека? Какие функции выносятся в библиотеку? В какой форме распространяются? Какие виды библиотек существуют, их преимущества. Как собрать динамическую библиотеку, особенности в зависимости от ОС. Как загрузить библиотеку в память? Как выгрузить?
39. Библиотеки. Динамические библиотеки. Подходы к реализации функций, которым необходимо создавать буфер динамически. Два способа: реализация функций выделения, освобождения памяти внутри библиотеки, перекладывание ответственности на вызывающую сторону (в таком случае рассказать о функциях, сообщающих вызывающей стороне необходимый размер буфера).
40. Библиотеки. Динамическая библиотека на Си. Приложение на Python (ctypes). Проблемы, которые возникают, когда библиотека и приложение на разных языках (загрузка, типы). Решение этих проблем с помощью ctypes.
41. Библиотеки. Динамическая библиотека на Си. Приложение на Python (модуль расширения). Проблемы, которые возникают, когда библиотека и приложение на разных языках (загрузка, типы). Решение этих проблем с помощью модуля расширения.
42. Неопределённое поведение. Побочный эффект. Пример выражения, анализ с двух сторон. Точки следования. Какие точки следования стандарт выделяет. Анализ выражения и борьба с неопределенным поведением с их помощью. Виды неопределенного поведения, примеры. Средства борьбы с неопределенным поведением.
43. Абстрактный тип данных. Понятие модуля. Разновидности модулей. Абстрактный объект "стек целых чисел". Преимущества модульной организации программ. Из каких частей модуль состоит, какие к ним требования? Какие средства существуют в языке Си для организации модулей? Что такое тип данных, АТД, чем они различаются? Пример реализации стека.
44. АТД. Понятие модуля. Разновидности модулей. Абстрактный тип данных «стек целых чисел».
45. Списки ядра Linux. Идеи, основные моменты использования. Что такое список ядра Linux, как достигается универсальность реализации? Как создать список, как добавлять элемент в начало/конец, как обойти список, как удалить элемент, как освободить память? Сравнение с обычными списками. При ответе на вопрос будет выдан распечатанный заголовочный файл list.h.
46. Списки ядра Linux. Идеи, основные моменты реализации. Что такое список ядра Linux? Как достигается универсальность? Какие обходы такого списка существуют, особенности (задача перехода от указателя на информационную часть к указателю на данные: containerof, offsetof).