На консультации (15/06/2002) Симоненко плохо отозвался по поводу этих вопросов-ответов, так как тут есть ошибки. Но я скажу, что по его выражению лица становится понятно, что ему не столько не нравятся ошибки, сколько то, что тут много правильного и мы получаем это на шару. Прим. Zerkella.

Раздел 1

1 **Математическое обеспечение.** совокупность методов и их описаний необходимых 4/ решения поставленной задачи.
2 **Многопрограммный режим работы** Режим, когда в системе находятся несколько задач в разной стадии исполнения, и каждая из них может быть прервана другой.
3 **Режим разделения времени** режим совмещающий мультипрограммирование и параллельную обработку, плюс возможность привелигированным пользователям иметь прямой доступ к ресурсам системы. (Логика прерываний)
4 **Классическое мультипрограммирование** Режим истинного совмещения, когда параллельно исполняемые задачи занимают различное оборудование.
5 **Параллельная обработка** Режим кажущегося совмещения, квантования.
6 Косвенный доступ. Доступ осуществляющийся через управляющую программу; предусматривает общение с вычислительной установкой косвенно.(telnet на сервер по сети) (Прообраз ОС, пакетный режим)
7 **Коллективный доступ** Предусматривающий доступ к ресурсам системы (система работает в многопрограммном режиме) многих пользователей, одновременная работа нескольких userov на машине (Логика прерываний)
8 Работа в реальном времени Режим реального времени – время реакции системы соответствует ранее заданному
9 **Перечислить способы реализации многопрограммного режима работы** разделение времени, классическое мультипрограммирование, параллельная обработка. Режим реального времени, Мультипрограммирование со свопингом.
10 **Классификация ОС** Однопрограммные, многопрограммные, однопроцессорные, многопроцессорные, распределённые, виртуальные.
11 **Особенности распределенных ОС(?)** Распределенная sys – совокупность выч. узлов, связанных между собой каналами связи, с точки зрения usera представляют собой единое целое.Отсутствие общей памяти приводит к невозможности определения общего состояния с помощью множества совместных переменных, а невозможность совместного обращения к памяти и различие в задержках передач сообщений приводит к тому что при определении состояния какого либо элемента системы из двух различных точек можно получить разные результаты. Выполнение работы распределяется в узлах исходя из соображения пропускной способности всей системы. Распределенные системы имеют высокий уровень организации параллельных вычислений.
12 **Основное отличие ОС UNIX (UNIX – Rulezzz!!!:)** ОС взаимодействует с аппаратурой непосредственно (программы-косвенно), обеспечивая обслуживание программ и их независимость от деталей аппаратной конфигурации.
13 **Дисциплины обслуживания заявок** Дисциплина обслуживания заявок – правило по которому решается задача обработки очереди заявок к каждому ресурсу. Бывают линейные (FiFo,Lifo,Random (нет одинаковых ресурсов)) циклические (RR,FBn,смешанная)
14 **Почему FIFO лучше LIFO** Дисциплина FiFo минимизирует дисперсию ожидания.
15 **Почему RR лучше FIFO** Длинная заявка не может захватить ресурс т.к., вводится понятие квант. Если заявка в RR не обслужена полностью, то по истечении кванта она помешается в очередь ; в FiFo заявка будет обслуживаться до ее выполнения.
16 **Почему FBn лучше RR.** Заявки в последней очереди выполняются без прерывания обслуживания => уменьшение временных затрат обусловленных прерыванием
17 **Почему алгоритм "Корбато" лучше FBn.** Алгоритм Корбато лучше алгоритма FBn, т.к. потенциально обладает большей производительностью т.к. распределяет задачи по очередям не по их приоритетам, а по признаку - абсолютная длина кода программы (чем меньше код тем выше приоритет)(В систему добавлен анализатор, который сразу размещает заявки в свою очередь, соответственно среднее время ожидания уменьшается.)
18 **Основная особенность приоритетных дисциплин обслуживания** Заявки имеют приоритет. Приоритетное обслуживание – заявка на вход системы с заданным приоритетом.
19 **Дать определение – область сохранения** Область сохранения - область в которую записывается информация о прерванном процессе, и откуда система берет данные при восстановлении процесса.
20 **Какая программа выполняет запись в область сохранения(?) –** Та часть ядра которая переключает задачи
21 **Виды модулей.** Виды: 1) исходный (текст программы); 2) объектный; 3) загрузочный; 4) исполняемый (абсолютный). Системные программы (компилятор, редактор связей, загрузчик) используются для преобразования от 1) к 4).
22 **Отличие загрузочного модуля от объектного.**Загрузочный модуль имеет все для своего выполнения . Написан на машинно-ориентированном языке, но быть выполненным не может. При использовании компилятора из исходного модуля получается объектный модуль, при использовании редактора связи из объектного получается загрузочный, далее из него при использовании загрузчика получается исполняемый модуль.
23 **Отличие загрузочного модуля от абсолютного.** Загрузочный модуль имеет все для своего выполнения . Написан на машинно-ориентированном языке, но быть выполненным не может. Нужно настроить адресные константы. Абсолютный – все адресные константы уже настроены.
24 **Дать определение – резидентный модуль.** модуль, который резидентно находится в памяти. (постоянно)Резидентные проги. – находятся в ОП, поддерживают быстрый отклик системы, Проги обработки прерываний, Процессы 4 / работы с ВУ, управление памятью, управление процессами, начальная прога. Управления заданиями.
25 **Дать определение – транзитный модуль.** Программы, связанные с выполнением функций ОС, но не находящиеся постоянно в ОП называются транзитными. Эти программы вызываются в ОП по мере не обходимости. – проги., которые могут понадобится 4/ выполнения ф-ций ОС (но не резидентные) – могут вызываться по оверлейной || динамически-последовательной схеме в транзитную зону.
26 **Связь модулей по управлению. Какие операции, какими программами выполняются.** Связь по управлению предусматривает сохранение состояния модуля на момент перехода на вызываемый модуль, причем действия по этому сохранению выполняет вызываемый модуль. Операции: возврат из i-1 модуля только в i+1; возврат из i+1 в i. Вызывающий модуль после передачи управления стирается. Виды программ могут быть : повторно не исполняемые, повторно исполняемые, чистые процедуры. (via common regs) Связь по управлению – связь для организации возврата. Возврат модуль В должен иметь команду, делающую так, что в области сохранения модуля А сохраняется состояние регистров. В модуле В есть адрес области сохранения в модуле А. Сохраняется адрес возврата, для возврата адреса следующей команды.
27 **Связь модулей по данным. Виды связей** Виды: по данным и по управлению. По данным может быть через общие регистры или через адрес списка параметром. Связь по данным – данные через общий регистр || через системный регистр

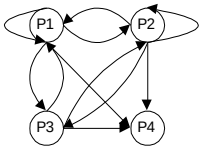
передается адрес списка параметров.

28 **Чем отличается традиционный машинный язык программирования от команд машины.** Команды машины - это самые элементарные выполняемые команды (самый низкий уровень); традиционный машинный язык - это набор процедур из элементарных команд, выполняемых на самом низком уровне.

29 **Назначения частей ОС ориентированной на пользователя и на 'hardware'.** Часть системы, ориентированной на «hardware» скрывает от пользователя те физические особенности системы, которые ему не нужны. (обеспечение взаимодействия устройств, процессов и т.д.). Часть ОС, ориентированной на пользователя, обеспечивает интерфейс с пользователем.

30 **Свойства модуля.** Стандартность внутренней структуры, функциональная завершенность, параметрическая универсальность, взаимная независимость. Если этот кусок имеет стандартную структуру, параметрическую универсальность, ф-циональную завершенность => u/ SUBJ.

31 **Показать все связи состояний P1 - P4.**



4 – состояния процесса – P1 – обработка системных и проблемных задач P2 – обработка прерываний P3 – дешифрация прерываний P4 – Exeption Переход на P3 при появлении сигнала прерывания, и выполнения дешифрации (определяется источник прерывания) – Реализовано на аппаратном уровне.

Раздел 2

32 **Виды планирования и из особенности** Динамическое (план составляется на том же оборудовании на котором выполняется решение задач во времени); статическое (план решения задач может составляться на другом оборудовании); балансовое (решение задачи балансирования нагрузки узлов в сети).Пространственный планировщик – решает задачу максимального паросочетания.

33 **Трудности планирования параллельных процессов** Добавляется планирование в пространстве; появляется проблема синхронизации процессов. ЕСЛИ в системе нет одинаковых устройств то очереди обрабатываются по линейной схеме => планирование только во времени ИНАЧЕ планирование во времени и в пространстве.

34 **Уровни планирования в однопроцессорной системе(?)** Заявка на входе ОС – задание – проходят через три уровня сверху (верхний,средний,нижний). Прошедшие верхний уровень претендуют на захват ресурсов в системе. Средний уровень обрабатывает приоритеты переводит задания в ранг задач. Планировщик среднего уровня активизирует ОС при наличии свободного ресурса. Задача – внутренняя единица работы системы, которой система выделяет ресурс. (д/ иметь Task Control Block) В каждой задаче имеется программа подчиненная задаче (обработка данных задачи). Планировщик нижнего уровня – определение очередности задач по выделению времени процессора.

35 **Дать определение и особенности динамического планирования** План составляется на том же оборудовании на котором выполняется решение задач во времени. Имеет ограничение на время составления плана. Динамическое планирование – задача планирования решается на том же оборудовании, что и выполняется план, который она составляет. Быстрое (чтобы не грузить оборудование) и неоптимальное решение Def. Джонсона – найти план распределения по ресурсам, при котором время решения не превышало бы критического с min кол-вом процов. Найти min кол-во ресурсов, чтобы задача решалась за min время. Для 2-х процов задача решается точно. Для одного в условиях RealTime.

36 **Уровни планирования в многопроцессорной параллельной системе (?)** Семиуровневая модель – схема системы планирования с учетом (Маштабируемость, Разделяемость, Параллельность) 1. Предварительное (входное) планирование исходного потока заявок (задача фильтрации) 2. Структурный анализ взаимосвязи входного потока заявок по ресурсам с определением общих ресурсов (Анализ) 3. Структурный анализ заявок и определение возможности распараллеливания (Задача распараллеливания) 4. Адаптация распределения работ соответственно особенностям ВС (Задача адаптирования) 5. Составление плана – расписания выполнения взаимосвязанных процедур. Оптимизация плана по времени решения и кол-ву ресурсов (Задача Оптимизации) 6. Планирование потока задач претендующих на захват времени процессора на каждый прцессор – задача распределения. 7. Выделение процессорного времени, активизация задач. Перераспределение работ в ВС, при отказе оборудования (задача распределения – перераспределения).

37 **Характеристика задач планирования с предшествованием(?)**

38 **Дать определение – двудольный граф -** имеющий только два уровня, связь между вершинами одного уровня невозможна.

39 **Алгоритмы динамического планирования, используемые в современных ОС(?)**

40 **Объяснить – поиск максимального парасочетания** Найти максимальное паросочетание - значит найти максимальное число ребер графа в которых не совпадают координаты вершин, либо найти максимальное число единиц матрицы у которых не совпадают координаты. Пространственный планировщик – решает задачу максимального паросочетания.

41 **Объяснить. Временная сложность алгоритма** Временная сложность – зависимость времени решения задачи от размерности задачи. (Берем наихудший случай)

42 **Чем отличается матрица связности от матрицы инцидентности** Инциндентности – 1 если вершина инциндентна ребру (столбец соответствующий ребру содержит ровно две единицы) Связности – если обе вершины (строка-столбец) имеют общие инцендентные ребра (инцендентные обоеим вершинам)

43 **Как определить минимальное кол-во процессоров, для загрузки распараллеленной задачи**

$$N_{\min} = \left\lceil \frac{\sum_{i=1}^{n} t_i}{T_{kp}} \right\rceil$$

NIow=]Sum(Ti)/Tkp[

44 **Способы определения сложности алгоритма(?)** аналитический и прогонка на тестовых примерах

45 **Задачи решаемые при статическом планировании** Поиск min-го кол-ва процессоров, необходимых для решения комплекса информационно- и по управлению взаимосвязанных задач за время, не превышающее заданное или критическое; поиск плана решения заданного комплекса информационно - и по управлению взаимосвязанных задач на заданном кол-ве процессоров за минимальное время. Def. Джонсона – найти план распределения по ресурсам, при котором время решения не превышало бы критического с min кол-вом процов. Найти min кол-во ресурсов, чтобы задача

решалась за min время.

46 **Основное отличие приоритетных дисциплин обслуживания** Приоритетное обслуживание – заявка на вход системы с заданным приопитетом. Относительный приоритет – не может прервать задачу на ресурсе, даже если она имеет низкий приоритет. Абсолютный приоритет – прерывает задачу на ресурсе если та имеет более низкий приоритет. Динамический приоритет – ЕСЛИ возникает опасность бесконечного откладывания. ЕСЛИ по мнению sys, заявка слишком долго занимает ресурс, ТО ее приоритет понижается. ЕСЛИ заявка очень долго ожидает ресурс ТО ее приоритет повышается.(QoS)

47 **Способы описания графа** Табличный и графический.

48 **Виды топологий параллельных систем(?)** PARA-PC Velsh – Рисунок – много ОЗУ много ПЭ – все подключены к облаку – Коммуникационная среда – (Статическая и динамическая коммутация(ПЭ-ПЭ или ПЭ ОЗУ)) => Можно моделировать (Общую Память) (Общая Шина) (КЭШ-шинно ориентированные – Общая шина + Каждый ПЭ общается с шиной via КЭШ (содержит программный код и промежуточные данные) (Система с виртуальной памятью – предидущее без ОП))Дерево Гипергуб

49 Как определить верхнюю границу кол-ва процессоров для погружения комплекса задач . Можно определить max число процессоров, выше которого практически не выгодно иметь больше процессоров – max ширина яруса.

50 **Классификация задач планирования(?)** Динамическое Статическое Балансовое В реальном времени

51 Критерии статического планирования Статическое планирование – план составляется на другой машине &&/|| в другое время. Оптимальный(оптимизированный) план за обозримое (физически нормальное время) Критерий оптимальности – время решения задачи и минимизация u/ресурсов. (NP-полная – времення сложность - експонента) Найти план решения рас||ной задачи представленной в виде ориентированного ациклического графа, время решения которого было бы равно Ткритическому с минимальным числом процов. 2. Найти мин. Время решения задачи при заданном кол-ве процов.

52 Как определить критическое кол-во процессоров для погружения программ(?)

53 Описать уровни статического планирования(?) Раннее Позднее

54 **Проблемы управления параллельными процессами** Добавляется планирование в пространстве; появляется проблема синхронизации процессов. Тупики

55 **Что такое граница Бременмана** Предел Бременмана(?) – if вычислительная сложность 2^63 => нужна машина с земной шар.

56 **Что такое увеличивающийся чередующийся путь(?)** Пространственный планировщик – решает задачу максимального паросочетания.(Алгоритм Карзанова и Карпа-Хопкрафта – Поиск максимального потока или увеличивающегося чередующегося пути)

57 **По какой характеристике можно судить о приемлемости разработанного алгоритма(?)** Временная сложность – зависимость времени решения задачи от размерности задачи. (Берем наихудший случай)

58 Задачи планирования в системах массового распараллеливания(?)

59 **Дать определение NP-полным задачам (**NP-полная – времення сложность – експонента – нельзя решить за полиномиальное время n^k) – для этой задачи не найдено полиномиальных алгоритмов но и не доказано что они не существуют

60 **Основные проблемы решения задач планирования в многопроцессорных параллельны системах(?) -** Добавляется планирование в пространстве; появляется проблема синхронизации процессов. Тупики

61 **Перечислить уровни планирования в классической ОС(?)** Заявка на входе ОС – задание – проходят через три уровня сверху (верхний,средний,нижний). Прошедшие верхний уровень претендуют на захват ресурсов в системе. Средний уровень обрабатывает приоритеты переводит задания в ранг задач. Планировщик среднего уровня активизирует ОС при наличии свободного ресурса. Задача – внутренняя единица работы системы, которой система выделяет ресурс. (д/ иметь Task Control Block) В каждой задаче имеется программа подчиненная задаче (обработка данных задачи). Планировщик нижнего уровня – определение очередности задач по выделению времени процессора.

62 Как можно использовать границу Бременмана(?) – Определить стоит ли использовать данный алгоритм

63 **Что такое задача назначения с предшествованием. Способы отображения(?)** Процесс планирования распред. ресурсов и задач ::= отображение Макрофакторов – 1.Системы (архитектура, структура, хак-ка) 2. Поток заданий (параметры заданий, требований, структуры) 3. Времени. – Выполнение заданий в пространственно временных координатах.Требование к задачам планирования работ по ресурсам – Нахождение отображения мн-ва ресурсов в мн-во работ исходя из условий оптимизации. (scheduling – без указания места расположения, mapping – с указанием места расположения) Полная задача статического планирования. Граф имеет веса вершин и пересылок . Надо решить все задачи о которых он говорил. W/u условие предшествования без прерывания задач, определить минимальное время и минимальное число процессоров.

64 **Принцип оптимальности Беллмана.(?)** Эвристические (основанные на опыте людей), Пошаговое конструирование – разделение всего процесса решения на определенное кол-во шагов и нахождение оптимального или квазиоптимального решения. ЭТО – соответствует принципу оптимальности Белмана.

65 **Сформулировать две классические задачи планирования в ОС(?)** Динамическое и статическое Требование к задачам планирования работ по ресурсам – Нахождение отображения мн-ва ресурсов в мн-во работ исходя из условий оптимизации. (scheduling – без указания места расположения, mapping – с указанием места расположения)

66 **Сформулировать цель решения задачи - "Нахождение максимального парасочетания"** Найти максимальное паросочетание - значит найти максимальное число ребер графа в которых не совпадают координаты вер шин, либо найти максимальное число единиц матрицы у которых не совпадают координаты.

Раздел 3

67 **Виды загрузчиков** Отличаются выполнением основных 4-х функций: распределения памяти, настройки, редактирования и загрузки. Загрузчики делятся: Абсолютный загрузчик, настраивающий загрузчик, непосредственно - связывающий загрузчик.

68 **Функции абсолютного загрузчика** Распределение памяти, настройка, редактирование;

69 **Функции настраиваемого загрузчика** Распределение памяти, настройка, редактирование, загрузка. Инфу для него готовит компилятор.

70 Какие топы загрузчика работают с модулями COM и EXE. С COM - абсолютный; с EXE - настраивающий.

71 **Дать определение. Генерация системы.** Процесс отбора из дистрибутива тех программных модулей, которые будут использоваться - т.е. создание операционной среды для конкретной конфигурации вычислительной системы. Из ДИСТИБУТИВА выбирают только те модули, которые поддерживают конкретную конфигурацию – ГЕНЕРАЦИЯ ОС.

72 **Дать определение. Резидентный том.** Место где находится резиденция системы называется резидентным томом, а сама система резиденцией.
73 Дать определение. Резиденция системы. РЕЗИДЕНЦИЯ – ОС полученная в результате генерации. Место нахождения резиденции – РЕЗИДЕНТНЫЙ ТОМ.
74 **Дать определение. Инициализация системы.** -процесс создания ядра системы который включает не только перезапись программ, но и системных структур, обеспечивающих знания системы о ее параметрах. Программа - загрузчик.
75 **Основные действия программы начальной загрузки** В MBR. Активизация программ начальной загрузки ОС.
76 Где находится программа начальной загрузки В MBR. (затем резидентный том)
77 Дать определение Табличный метод управления Метод когда решение управляющей системы принимается на основании инфы, хранящейся в таблицах, содержащих данные состояния всех частей системы.
78 **Почему используются многоуровневые системы программирования** Для облегчения разработки компилятора (ставится несколько уровней трансляторов). Декомпозиция – разбиение сложной задачи на более мелкие. (Каждый класс задач на своем уровне связь вниз – выполнение ф-ции, вверх возврат результата) Стандартизация интерфейса – удобство собирать по кирпичикам
79 **Какие программы находятся в ядре ОС (Виды программ)** ядро супервизора, включающие только те программы, которые отвечают за реакцию системы.
80 **Функции библиотекаря** Работа с библиотекой : поиск, запись, удаление, редактирование, копирование, запись без каталогизации.
81 **Что такое временная библиотека** Библиотека записанная без каталогизации.
82 **Структура библиотеки.** Сама библиотека, управляющая программа, каталог.
83 **Что такое запись в библиотеку без каталогизации** Запись временных данных, которые удаляются после выполнения программы.
84 Особенности перемещения (ориг. стирання) программы из библиотеки на внешний носитель.(?)
85 Дать определение – программа простой структуры Программа которая при её исполнении не обращается к другим программам. Простейшая структура (имеют в своем теле все что нужно для выполнения)
86 Дать определение – программа оверлейной структуры -Задача, разделяемая на модули. Модули находящиеся на одном оверлейном уровне не могут одновременно находится в ОП. Оверлей (заданные перекрытия два модуля на одном оверлейном уровне не могут быть выполнены Инфа передается via корневой модуль)
87 Дать определение – программа динамически последовательной структуры. Программы настроенные с помощью модульного принципа, представляется в перемещаемом виде, могут подгружаться по мере необходимости с организацией связей по управлению и данными. 3. Динамически посл.(Память выделяется по мере надобности – по цепочке. Уровни не выделяются. Корневой модуль всегда в памяти)
88 **Дать определение – программа динамически параллельной структуры.** Динамически - последовательная структура с указанием о том какие модули могут выполняться параллельно
89 Как определить объем памяти необходимой для загрузки программы обычной структуры. Простой - размер самой программы + корневой каталог.
90 **Как определить объем памяти необходимой для загрузки программы оверлейной структуры.** Оверлейной - Корневой сегмент + самый большой модуль.
91 **Как определить объем памяти, для загрузки программы динамически параллельной структуры.** Корневой сегмент + наибольший суммирующий объем модулей, которые могут выполнятся параллельно. Какая системная программа подготавливает информацию для работы загрузчика. Компилятор. (###редактор связей) Непосредственно связывающие – инфу – для них готовит компилер. (в спец. виде.)
92 **Какую информацию и как компилятор передает настраивающему загрузчику** Настраивающий – работает в загр. Модуле которому больше ничего не надо. Постепенно его ф-ции взял на себя редактор связей. Непосредственно в настраивающем загрузчике каждый модуль может транслироваться отдельно. Чтобы передать сообщение редактору связей надо ему непосредственно указать, что надо транслировать. В каждом модуле в начале трансляции выделяются вектора перехода, внешние и внутренние.(Экспорт и Импорт процедур и ф-ций). Кроме того для выполнения настройки каждая команда отмечается битом переместимости. ОС выделяет и пользуетс глобально выделенной памятью, а загрузчик с локальной.
93 **Функции редактора связей** Редактирование связей.(Выполнение связывания подпрограмм являющихся внешними по отношению к загружаемому модулю). (С помощью редактора связи мы получаем из объектного модуля загрузочный модуль имеющий всё для своего исполнения).
94 **Выходная и входная информация редактора связей** Входная - объектный модуль, выходная - загрузочный модуль.
95 **Определение – что есть прерывание** -событие, требующее от системы прекращение работы активного процесса и перехода к обработке другого задания.
96 Что такое система прерываний(?) Комплекс программно-аппаратных средств, обеспечивающих прохождение сигнала прерывания от устройства к части ядра отвечающей за обслуживание прерывания.
97 Что такое система обслуживания прерывания Комплекс программно-аппаратных средств, обеспечивающих приостановку активного процесса и переход к обработке прерывания.
98 **Классы прерываний** От схем контроля машиной; внешние; прерывания по вводу/выводу; по обращению к супервизору; программные/
99 **Виды PSW.** Текущий (в регистрах процессора), новый (в векторах прерываний), старый (в постоянной области памяти вместе с векторами прерываний; в стеке той программы которая прерывается или в PCB прерванного процесса).
100 **Информация, которая хранится в PSW.** Уникальный идентификатор процесса (имя); текущее состояние процесса; приоритет процесса; указатели участка памяти выделенного программе, подчиненной данному процессу; указатели выделенных ему ресурсов; область сохранения регистров; права процесса (список разрешенных операций); связи зависимости в иерархии процессов (список дочерних процессов, имя родительского процесса); пусковой адрес проги, подчиненной данному процессу. А также дополнительная инфа для синхронизации процессов, 4 поля : 1-2: поля для организации цепочки связи; 3-4: поля для организации цепочки ожидания.
101 **Сеанс смены PSW.** PSW служит для реализации переключения процессов. В случае сохранения старых PSW в постоянной области памяти глубина определяется классов прерываний. Если старые PSW сохраняются в стеке то глубина определяется размером памяти отведенной под стек.
102 Где находятся старые PSW. старый (в постоянной области памяти вместе с векторами прерываний; в стеке той программы которая прерывается или в PCB прерванного процесса). Старые PSW могут хранится в в постоянной области памяти вместе с векторами прерываний; в стеке той программы которая прерывается или в PCB прерванного процесса.

103 Где находятся новые PSW. Значение нового PSW находится в векторе прерывания с номером n, где n указывает номер прерывания обработчик которого нужно взять. новый (в векторах прерываний)

104 Зачем используется CAW. (?)

105 **Почему и когда блокируется система прерываний.** При дешифрации текущего прерывания

106 **Обработка программных прерываний** Выполняемый процесс.вырабатывает Принимает Ядро ОС

107 Где фиксируется прерывание ввода-вывода(?) В контролере i/o и в КПП

108 **Кто вырабатывает сигнал прерывания по вводу-выводу** Аппаратура В/В при изменении состояния каналов В/В, а также при завершении операций В/В.

109 Кто вырабатывает сигнал прерывания по запросу к супервизору Выполняемый процесс.

110 От чего зависит момент прерывания(?) Х3 от чего

111 **Из чего состоит время ожидания при обработке сигнала прерывания** Время на сохранения состояния прерываемой задачи; время на сохранения адреса возврата; загрузка вектора прерываний.

112 **Что такое маскирование прерываний** После того как прерывание замаскировано контроллер игнорирует появление сигнала этого прерывания. Осуществляется путем загрузки регистра маски КПП.

113 **Что такое зацикливание приоритетов(?)** Бесконечное откладывание – решение динамические приоритеты

114 Что такое насыщение системы прерываний(?)

115 **В чем различие между утилитами и сервисными программами** Нет никакой разницы. Утилита это и есть сервисная программа.(другой вариант: утилиты входят в состав сервисных программ).

116 **Дать определение – обрабатывающие программы ОС** Программы выполнения стандартных (в рамках ОС) функций, обработки исключительных ситуаций. Обработка – изменение ин-фы с которой работает прога в составе ОС (дрова, загрузчик)

117 **Дать определение – управляющие программы ОС** Программы постоянно находящиеся в памяти (резидентные) организующие корректное выполнение процессов и функционирование всех устройств системы при решении задач. Составляют ядро ОС. Управление : Заданиями – слежение за прохождением заданий от входа до выхода на всех этапах его выполнения. Задачами – (Процессами) – слежение за всеми задачами активизированными в системе и процессами их выполнения на ресурсах. Памятью – решение задач эффективного u/ mem (internal) в соответствии с ее организацией.(Защита – согласование ин-фы в кешах) Данными – эффективное размещение и u/ данных на внешних носителях (проблема эффективности u/ процессора) Внешними ус-вами …

118 **Что такое спулинг** Режим буферизации для выравнивания скоростей при вводе и считывании информации из буфера.При работе программ системного ввода u/ режим спуллинга – согласование скоростей на входе и выходе.

119 **Что такое свопинг** Свопинг - способ реализации многопрограммного режима работы на однопроцессорной машине.Проблема настройки связана с перемещением программ в ОП, ЕСЛИ ОС работает со свопингом – нужна перенастройка адресных констант. (глобальная – перенастройка всех адресных констант, локальная – вычисление адреса той переменной, которая находится реально в ОП)

120 **Основное отличие компилятора от интерпретатора** Интерпретатор выполняет перевод части проги в машинные команды и тут же ее выполняет, выполняя все необходимые настройки адресных констант. Компилятор создает объектный модуль, который затем обрабатывается редактором связей. Во время работы интерпретатора исполняемый код программы записывается в фиксированное место и управление передается на стартовый адрес программы.

121 **Что такое виртуальная ОС** Так называемая ОС, которая позволяет многим пользователям работающим на одной и той же технической базе(одно ус-во) одновременно работать в различных операционных средах. Виртуальные ОС – разрешает нескольким useram работать на одном и томже dev на разных ОС.

122 **Перечислить классы прерываний по возрастанию приоритета** От схем контроля машины, внешние, по вводу/выводу, по обращению к супервизору, программные.

123 **Какая системная программа готовит команду начать ввод-вывод.** Обработчик прерываний.

124 Какая часть ОС обрабатывает сигналы прерываний Ядро.

125 **В чем сложность для ОС организации многопрограммного режима работы** Организация защиты от взаимного влияния друг на друга на уровне оперативной и на уровне внешней памяти; разделение аппаратных и программных ресурсов; планирование (во времени, а в случае ПВС и в пространстве).

126 **Что такое прямой ввод-вывод** Когда не используются управляющие программы. Пользовательский процесс сам осуществляет i/o

127 Что такое псевдонумерация прерываний(?) Номер не означает приоритет

128 **Дать определение – процесс** Это любая выполняемая работа в системе; это динамический объект (внутренняя единица работы) системы, которому она выделяет ресурсы;

129 **Условие перехода процесса из подготовительного состояния в готовность.** Выделение ресурсов.

130 **Условие перехода процесса из состояния готовности в активное.** Выделение кванта времени процессора, запуск.

131 **Условие перехода процесса из активного состояния в блокированное.** Ожидание события, освобождения ресурса, конца операции ввода/вывода.

132 **Условие перехода процесса из блокированного состояния в готовое.** Наступление события, освобождения ресурса, завершение ввода/вывода.

133 **Условие перехода процесса из активного состояния в готовое.** Истечение кванта времени. (Выделенное время процессора для выполнение этого процесса завершилось и процессор занят другим процессом).

134 **Условие создания процесса.** Корректность описания процесса на языке описания процессов и выделение ресурсов

135 **Отличие создания процесса при рекурсивном обращении (?)** Хранить все локальные переменные в стеке.