1. **Почему однопрограммный пакетный режим – более производительный, чем индивидуальный (однопрограммный, надо понимать)?**

   Хорошо иллюстрирует пример Игоря Викторовича про перфокарты. Когда система находится во владении одного пользователя, в случае её отказа или сбоя ресурсы системы простаивают, пока пользователь отлаживает и думает. В пакетном же режиме в случае отказа запускается следующая задача - система не простаивает. Особенно это было актуально понятно когда.

2. **Какой добавочный резерв производительности задействует пакетный мультипрограммный режим по сравнению с пакетным однопрограммным?**

   Устройства ввода-вывода — можно считать и печатать одновременно, например. Да и вообще разделяемость ресурсов (процессор, память).

3. **Какой из режимов обеспечивает самую высокую производительность? Реактивность?**

   Реактивность - реального времени. Производительность - пакетный (или индивидуальный?).

4. **В вычислительных центрах существует многопользовательский режим удаленного ввода заданий (Remote Job Entry, RJE), промежуточный между 2.1 и 2.2. Он эффективен, когда несколько программистов работают в цикле подготовка/отладка программ. Охарактеризуйте его.**

   Людей имеют доступ к ресурсам, которые по некоторым причинам не воспроизводимы на ПК. Похож на мультипрограммный пакетный. Как охарактеризовать?..

5. **Как размер кванта времени в режиме разделения времени влияет на реактивность? На производительность?**

   Чем меньше размер кванта, тем меньше реактивность, очевидно, но также тем меньше задач можно выполнить в течение такого кванта - производительность ниже.

6. **В чем преимущества однопользовательского мультипрограммирования W2k по сравнению с однопрограммным режимом, единственно возможным в MS-DOS?**

7. **В Win2k UniqueProcessId (сокр. PID) – целые, кратные 4. Зачем?**

   Побочный эффект повторного использования кода. Тот же код использовался для выделения дескрипторов в ядре, а вот там они делятся на четыре, потому что два последних бита всегда ноль, делается для всяких разных целей.
   `http://blogs.msdn.com/b/oldnewthing/archive/2008/02/28/7925962.aspx`

8. **Значительную долю времени переключения контекста занимает перегрузка содержимого общих регистров в PCB и обратно. Как можно этого избежать – ценой увеличения объема аппаратуры процессора (что сделано, в частности, в архитектуре RISC–процессоров Sun Sparc) ?**

   Большое количество регистров общего значения, разбитой на небольшие наборы (регистровые окна). Так вроде в Sun Sparc и сделано.

   The SPARC processor usually contains as many as 160 general purpose registers. At any point, only 32 of them are immediately visible to software - 8 are a set of global registers (one of which, g0, is hard-wired to zero, so only 7 of them are usable as registers) and the other 24 are from the stack of registers. These 24 registers form what is called a register window, and at function call/return, this window is moved up and down the register stack. Each window has 8 local registers and shares 8 registers with each of the adjacent windows. The shared registers are used for passing function parameters and returning values, and the local registers are used for retaining local values across function calls.

   The "Scalable"in SPARC comes from the fact that the SPARC specification allows implementations to scale from embedded processors up through large server processors, all sharing the same core (non-privileged) instruction set. One of the architectural parameters that can scale is the number of implemented register windows; the specification allows from 3 to 32 windows to be implemented, so the implementation can choose to implement all 32 to provide maximum call stack efficiency, or to implement only 3 to *reduce context switching time*

9. **Приведите пример «местоимения» при именовании файловых каталогов.**

   /Documents and settings/fzentsev/*My* documents. А может и нет, не зря в кавычках дано.

10. **Приведите пример контекста сеанса пользователя Unix или Windows.**

    Предположение: /home/username в Unix, Desktop в Windows. А может и нет. Может список в ProcessExplorer или htop - это контекст сеанса?

11. **Как принцип умолчания позволяет кратко именовать файлы в иерархических каталогах? Приведите другие примеры принципа умолчания в пользовательских интерфейсах.**

    /work/company1projects/projectname/projectcore/*src*