26. Стек и куча.

План ответа:
1 автоматическая память: использование и реализация;
2 использование аппаратного стека (вызов функции, возврат управления из функции, передача параметров, локальные переменные, кадр стека);
3 ошибки при использовании автоматической памяти;
??? 4 динамическая память: использование и реализация; ???
5 идеи реализации функций динамического выделения и освобождения памяти ???

1 Автоматическая память используется для хранения локальных переменных
+ Память под локальные переменные выделяет и освобождает компилятор.
- Время жизни локальной переменной "ограничено" блоком, в котором она определена.
— Размер размещаемых в автоматической памяти объектов должен быть известен на этапе компиляции.
— Размер автоматической памяти в большинстве случаев ограничен.

2 Аппаратный стек используется для:
1. вызова функции
call name
поместить в стек адрес команды, следующей за командой call
передать управление по адресу метки name
2. возврата из функции
ret
извлечь из стека адрес возврата address
передать управление на адрес address
3. передачи параметров в функцию
соглашение о вызове:
расположение входных данных;
порядок передачи параметров;
какая из сторон очищает стек;
cdecl
аргументы передаются через стек, справа налево;
очистку стека производит вызывающая сторона;
результат функции возвращается через регистр EAX, но …

4. выделения и освобождения памяти под локальные переменные

Стековый кадр (фрейм) - механизм передачи аргументов и выделения временной памяти с использованием аппаратного стека.
В стековом кадре размещаются:
значения фактических аргументов функции;
адрес возврата;
локальные переменные;
иные данные, связанные с вызовом функции.
+ Удобство и простота использования.
— Передача данных через память без необходимости замедляет выполнение программы.
— Стековый кадр перемежает данные приложения с критическими данными - указателями, значениями регистров и адресами возврата.

3 Ошибки:
возврат укателя на локальную переменную
переполнение буфера

4-5 ??? Динамическая память
Куча представляет собой непрерывную область памяти, поделённую на занятые и свободные области (блоки) различного размера.
Информация о свободных и занятых областях кучи обычно храниться в списках различных форматов.
При запуске процесса ОС выделяет память для размещения кучи.

Алгоритм работы malloc
просматривает список занятых/свободных областей памяти, размещённых в куче, в поисках свободной области подходящего размера;
если область имеет точно такой размер, как запрашивается, добавля

ет найденную область в список занятых областей и возвращает указатель на начало области памяти;

если область имеет больший размер, она делится на части, одна из которых будет занята (выделена), а другая останется в списке свободных областей;

если область не удается найти, у ОС запрашивается очередной большой фрагмент памяти, который подключается к списку, и процесс поиска свободной области продолжается;

если по тем или иным причинам выделить память не удалось, сообщает об ошибке (например, malloc возвращает NULL).

Алгоритм работы free

просматривает список занятых/свободных областей памяти, размещённых в куче, в поисках указанной области;

удаляет из списка найденную область (или помечает область как свободную);

если освобожденная область вплотную граничит со свободной областью с какой-либо из двух сторон, то она сливается с ней в единую область большего размера.