16. Динамические одномерные массивы.

План ответа:
	1 Функции для выделения и освобождения памяти (malloc, calloc, realloc, free). Порядок работы и особенности использования этих функций.
	2 Типичные ошибки при работе с динамической памятью (утечка памяти, «дикий» указатель, двойное освобождение).

1 Для выделения памяти необходимо вызвать одну из трех функций, объявленных в заголовочном файле stdlib.h:
– malloc (выделяет блок памяти и не инициализирует его);
– calloc (выделяет блок памяти и заполнят его нулями);
– realloc (перевыделяет предварительно выделенные блок памяти).

Особенности:
	Указанные функции не создают переменную, они лишь выделяют область памяти. В качестве результата функции возвращают адрес расположения этой области в памяти компьютера, т.е. указатель.
	Поскольку ни одна из этих функций не знает данные какого типа будут располагаться в выделенном блоке все они возвращают указатель на void.
	В случае если запрашиваемый блок памяти выделить не удалось, любая из этих функций вернет значение NULL.
	После использования блока памяти он должен быть освобожден. Сделать это можно с помощью функции free.

Результат вызова функций malloc, calloc или realloc, когда запрашиваемый размер блока равен 0, зависит от реализации (implementation-defined C99 7.20.3):
– вернется нулевой указатель;
– вернется «нормальный» указатель, но его нельзя использоваться для разыменования.
ПОЭТОМУ перед вызовом этих функций нужно убедиться, что запрашиваемый размер блока не равен нулю.

malloc
	void* malloc(size_t size);
		Функция malloc выделяет блок памяти указанного размера size. Величина size указывается в байтах.
		Выделенный блок памяти не инициализируется (т.е. содержит «мусор»).
		Для вычисления размера требуемой области памяти необходимо использовать операцию sizeof.
	a = (int*) malloc(n * sizeof(int));
		Преимущества явного приведения типа:
			– компиляции с помощью c++ компилятора;
			– у функции malloc до стандарта ANSI C был другой прототип(char* malloc(size_t size));
			– дополнительная «проверка» аргументов разработчиком.
		Недостатки явного приведения типа:
			– начиная с ANSI C приведение не нужно;
			– может скрыть ошибку, если забыли подключить stdlib.h;
			– в случае изменения типа указателя придется менять и тип в приведении.

calloc
	void* calloc(size_t nmemb, size_t size);
		Функция calloc выделяет блок памяти для массива из nmemb элементов, каждый из которых имеет размер size байт.
		Выделенная область памяти инициализируется таким образом, чтобы каждый бит имел значение 0.

realloc
	realloc(void* ptr, size_t bytes) перевыделяет память под объект ptr. Выделяют следующие случаи:
		ptr==NULL && bytes>0. Происходит обычное выделение памяти, как при malloc(bytes).
		ptr!=NULL && bytes==0. Происходит освобождение памяти, как при free(ptr).
		ptr!=NULL && bytes!=0. В худшем случае выделяется bytes байтов, к

опируются имеющиеся значения байтов из старой области памяти в новую, освобождается старая память.
В лучшем случае, когда соседние справа байты свободны в достаточном количестве или bytes не больше текущего размера выделенной области, перемещений не происходит.

Особенность использования: отслеживать, если realloc возвращает NULL (результат записать в отдельный буфер).

```
void *ptmp = realloc(pbuf, 2 * n);
if (ptmp)
        pbuf = ptmp;
else
        // обработка ошибочной ситуации
```

free

```
void free(void *ptr);
```

Функция free освобождает (делает возможным повторное использование) ранее выделенный блок памяти, на который указывает ptr.
Если значением ptr является нулевой указатель, ничего не происходит.
Если указатель ptr указывает на блок памяти, который не был получен с помощью одной изфункций malloc, calloc или realloc, поведение функции free не определено.

2 Типичные ошибки

- Утечки памяти (memory leak)
- Разыменование «битого» указателя(invalid pointer)
- Двойное освобождение памяти(double free)