alloc.h

```
#ifndef ALLOC_H_INCLUDE
#define ALLOC_H_INCLUDE

#include <stdlib.h> // size_t
#include <stdint.h> // int32_t

/**
 * memory test malloc
 * This function shouldn't be called directly, but rather the macros below used
 * this will supply them with tracing information
 * when MEMTEST is defined, mt_malloc performs an allocation with padding,
 * the allocation is stored internally and a pointer that can be used nas
 * normal is returned.
 * If MEMTEST isn't defined, just does malloc from stdlib.h
 * @param sz how much memory to allocate
 * @param file the file this function is called from
 *            (should be filled with __FILE__).
 * @param line the line this function is called from
 *            (should be filled with __LINE__).
 */
void* mt_malloc_(const size_t sz,
            const char* file, const size_t line);

/**
 * memory test calloc
 * This function shouldn't be called directly, but rather the macros below used
 * this will supply them with tracing information
 * when MEMTEST is defined, mt_calloc performs an allocation with padding,
 * the allocation is stored internally and a pointer that can be used nas
 * normal is returned.
 * If MEMTEST isn't defined, just does calloc from stdlib.h
 * The allocated memory has it's value set to 0, just like calloc
 * @param n  how many elements to allocate for
 * @param sz how big each element is
 * @param file the file this function is called from
 *            (should be filled with __FILE__).
```

```
 * @param line the line this function is called from
 *             (should be filled with __LINE__).
 */
void* mt_calloc_(const size_t n, const size_t sz,
                 const char* file, const size_t line);

/**
 * memory test free
 * works just like free(). When MEMTEST is defined it should only be used on
 * pointers allocated with the memory test functions, calling on memory that
 * wasn't allocated by a memory test function will cause an abort.
 * This function will print any under or overwrites to stderr.
 * @param p pointer to free
 */
void mt_free(void* p);

/**
 * An internal function to check details and clean up of any leaked memory.
 * It is called automatically on exit via atexit()
 * When MEMTEST isn't defined, has no effect
 */
void mt_check();

#define mt_malloc(x) mt_malloc_(x, __FILE__, __LINE__)
#define mt_calloc(x) mt_calloc_(x, __FILE__, __LINE__)
#define mt_realloc(x) mt_realloc_(x, __FILE__, __LINE__)

#endif
```