Programming assignment

Integration engineer - platforms area

Contact info or
something like that idk

More unnecessary details and stuff Yeah more here

Introduction

Warioware™ makes software which enables the creation of wonderful interfaces for our Mario-karts. Our Bananzi® software is used by leading kart manufacturers.

The purpose of this assignment is to evaluate your ability to work on programming tasks resembling daily activities of the Warioware R&D department.

This focuses on cross-platform programming on embedded systems using C++.
We evaluate the quality of both code and documentation, as well as the actual production use applicability of your implementation.

The R&D department regards these coding characteristics as valuable:

- Readability & simplicity
- Reliability & safety
- Modularity & maintainability
- Efficiency & aptness

Instructions

Write a C++ library implementing a memory allocator for embedded systems.
The requirements for the memory allocator are as follows:

1. The memory allocator will have a well defined state that does not change outside calls to its functions.

2. The success or failure of a call to a function shall only depend on the current state of the memory manager, and the arguments of the function call.

3. Given a C++ type T, and an ordered set of arguments compatible with a constructor for type T, the memory manager will attempt to allocate a memory block large enough to hold an object of type T aligned at the natural alignment for type T, then construct an object of type T inside that memory block and return a pointer to the object, if possible.

4. The memory manager will be able to destruct objects it constructed and deallocate valid memory blocks. Such a valid deallocation may never fail. A memory block is considered valid, if it was previously allocated using the memory allocator and has not since been deallocated.

5. If the user makes a sequence of successful allocations starting from any state X and subsequently deallocates that memory, the memory manager shall end up in state X', from where that same sequence of memory allocations is guaranteed to succeed. In other words, the states X and X' shall be indistinguishable for the user. This requirement shall apply regardless of the order in which the blocks were deallocated.

6. If the user makes a failed allocation starting from any state Y, the memory allocator will end up in state Y', from where that same memory allocation is guaranteed to fail. In other words, the states Y and Y' shall be indistinguishable for the memory manager user.

7. These requirements will be met even when the underlying system implementation does not guarantee them.

The performance requirements for the memory allocator are as follows:

1. In a state with n valid allocations, an allocation will complete in O(log n) steps amortized. However, it shall always complete in O(n) steps.

2. In a state with n valid allocations, a deallocation will complete in O(log n) steps.

The requirements for the C++ programming library are as follows:

a) The implementation language is C++14 or C++17.

b) The library only depends on the C++ Standard Library and POSIX.

c) Each feature of the library shall be documented.

d) Each function shall document:
    1. The intended behavior.
    2. All function parameters.
    3. Return value(s).
    4. All failure modes.

e) The library includes enough tests to demonstrate that the memory manager's requirements are met.

f) The library includes instructions to build and execute the tests on a recent version of Ubuntu LTS (x86_64).

Submission

Provide your solution in a .zip archive, which includes the project files.

The documentation will include a brief description of your implementation, the tools you used, and possible known issues. If you used AI or copied code from others, identify the sources and specify your own contribution.