Princeton University
COS 217: Introduction to Programming Systems
Heap Manager: Algorithms for Second Assignment Implementation

void *HeapMgr_malloc(size_t uBytes)

(1) If this is the first call of HeapMgr_malloc(), then initialize the heap manager.

(2) Determine the number of units the new chunk should contain.

(3) For each bin from the proper start bin to the last bin...

    For each chunk in the current bin...

        If the current chunk is big enough...

            If the current chunk is close to the requested size, then remove it from its bin, set its status to INUSE, and return it. If the current chunk is too big, then remove it from its bin, split the chunk, insert the **tail** end of it into the proper bin at the front, set the status of the **front** end of it to INUSE, set the status of the **tail** end of it to FREE, and return the **front** end of it.

(4) Ask the OS for more memory — enough for the new chunk. Return NULL if the OS refuses. Create a new chunk using that memory. Insert the new chunk into the proper bin at the front. If appropriate, coalesce the new chunk and the previous one in memory. To do so, remove the current chunk from its bin, remove the previous chunk in memory from its bin, coalesce the two chunks to form a larger chunk, and insert the larger chunk into the proper bin at the front. Let the current chunk be the new chunk.

(5) If the current chunk is close to the requested size, then remove it from its bin, set its status to INUSE, and return it. If the current chunk is too big, then remove it from its bin, split the chunk, insert the **tail** end of it into the proper bin at the front, set the status of the **front** end of it to INUSE, set the status of the **tail** end of it to FREE, and return the **front** end of it.

void HeapMgr_free(void *pv)

(1) Set the status of the given chunk to FREE.

(2) Insert the given chunk into the proper bin at the front.

(3) If appropriate, coalesce the given chunk and the next one in memory. To do so, remove the given chunk from its bin, remove the next chunk in memory from its bin, coalesce the two chunks to form a larger chunk, and insert the larger chunk into the proper bin at the front.

(4) If appropriate, coalesce the given chunk and the previous one in memory. To do so, remove the given chunk from its bin, remove the previous chunk in memory from its bin, coalesce the two chunks to form a larger chunk, and insert the larger chunk into the proper bin at the front.