Lab2 report

Task1：

Default\_init\_memmap中将base开始的n页主意初始化，property和flag置为0，将base的property置为n，将其根据地址大小依次加入free\_list

Default\_alloc\_page中从free\_list起始开始找符合大小的块，找到后取出对应的大小将其中各个页标记为已经取出使用，并从freelist移除，剩下的部分（如果有）设置为新的free块（改变property）。

Default\_free\_page中将释放的块重新添加到freelist中所在位置（根据base的地址），并且重新设置base，将property置为n。依次和前面一个、后面一个块比较，如果它们恰好相邻就将他们合并。

我的实现最有机会优化的地方是Default\_alloc\_page中从free\_list起始开始找符合大小的块，这里是从头到尾扫描，可以加入数据结构的优化将这个寻找的过程复杂度从n降到logn（比如线段树）。

Task 2:

找到la对应的页目录表项，根据该项内容得知la对应的页表地址，再又la的数据得到需要返回找到页表的第几项。当la的页表项不存在，如果不需要新建页表项或者新建页表项失败就返回null，否则建立新页表项，设置该项被访问了一次，将该项内容初始化（清空），根据该项地址重新设置相应页目录项的值。

页目录项最后三位是标记位，从低位到高位分别表示是否存在，是否可写，是否能被用户访问，页目录项其余位记录页表地址，页表项最后两位是标记位，从低位到高位分别表示是否存在，是否可写。页表项其余位记录对应的内存地址。这些标记位对于系统管理内存保护数据分隔核心层用户层有作用。

当异常出现时，硬件将向从外界存储设备发出信号，外界存储设备和硬盘中交互数据。

Task 3:

当传入的页表项存在时，找到页表项对应的页，将其引用次数减1如果引用次数变成零就释放该页。将页表项清零，然后让tlb更新。

有对应关系，页表是从end开始的连续一段内存存放的。

令gcc编译出的虚拟起始地址从0x100000开始即可。

知识点：

缺页相关的知识点在实验中没有体现