实验二：伙伴算法代码纠错

姓名：潘梓月 学号：22551192 指导老师：赵新奎 实验日期：2025/10/27

该代码片段为 Unikraft 操作系统的 `bbuddy_pfree` (buddy allocator page free) 函数。经分析，代码中存在以下2处主要问题：

问题一：Buddy System 合并逻辑被禁用

代码中使用了 `#if 0` 预处理器指令，将核心的 "buddy" (伙伴) 合并逻辑（coalescing logic）完全禁用了。

取而代之的 `else` 分支中的逻辑是错误的。它将一个（可能很大的）内存块（例如一个 8 页的 order-3 块）拆散成 2^{order} 个单独的 order-0 页面（$2^3 = 8$ 个 1-page 块），并将它们逐个添加到 `free_head[0]` 链表中。

这完全违背了 Buddy System 的设计初衷。Buddy Allocator 的核心优势在于**合并**相邻的空闲块以形成更大的可用内存块，从而减少内存碎片。当前激活的 `else` 逻辑会导致严重的内存碎片化，使得系统在释放大块内存后，也无法再分配出同样大小的内存块。

纠正方法： 启用正确的合并逻辑，将 `#if 0` 修改为 `#if 1`。

问题二：链表操作中存在 NULL 指针解引用风险

在（被禁用的）`#if 0` 块中，有两处对双向链表的操作存在风险，它们在访问 `next` 节点的 `pprev` 指针前，没有检查 `next` 节点是否为 `NULL`。

1. **合并时解链 (Unlinking)**:

   ```
   *(to_merge_ch->pprev) = to_merge_ch->next;
   to_merge_ch->next->pprev = to_merge_ch->pprev; // <-- 问题点 1
   ```

   如果 `to_merge_ch` 恰好是其所在 `free_head[order]` 链表的最后一个节点，那么 `to_merge_ch->next` 将为 `NULL`。此时 `NULL->pprev` 会导致段错误 (Segmentation Fault)。

2. **链接新块 (Linking)**:

   ```
   freed_ch->next = b->free_head[order];
   ...
   freed_ch->next->pprev = &freed_ch->next; // <-- 问题点 2
   b->free_head[order] = freed_ch;
   ```

   如果 `b->free_head[order]` 链表为空（即 `freed_ch->next` 被赋值为 `NULL`），那么 `NULL->pprev` 同样会导致段错误。

纠正方法： 在这两处赋值操作前，增加对 `next` 指针的 `NULL` 检查。