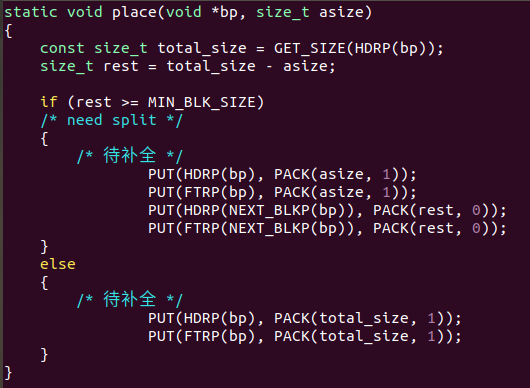
隐式空闲链表：

**static void \*ﬁnd\_ﬁt(size\_t asize)：**

![C:\Users\Lucifer.dark\Documents\Tencent Files\709990727\Image\C2C\BPPX)](45%U{9Z9PJQ5(BQG.png](data:image/png;base64,)

此函数用于寻找合适的空闲块，从头开始找，所找的块应该是未分配，且块大小需要大于等于所需要的大小。块的大小及是否已被分配存在头部、脚部，所以利用GET\_SIZE(HDRP(bp))就可以获得块的大小。用GET\_ALLOC(HDRP(bp))就可以知道块是否已被分配。利用NEXT\_BLKP(bp)即可获得bp所指向块的相邻下一块的地址。如果当前存在的空闲块无合适的，就返回NULL。(无合适块就会扩展空间)

**static void place (void \*bp, size\_t asize)：**

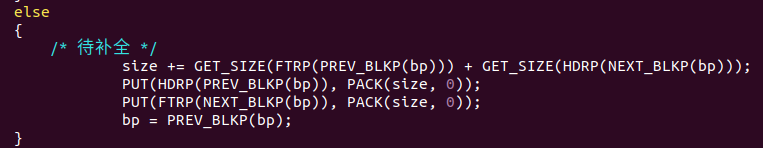


此函数用于判断找到的空闲块大小是否大于请求的内存大小，。如果找到的块的大小减去所需块的大小大于等于4个字节的大小（独立空闲块所需最小的大小），就进行切割，否则不切割（即使有内部碎片）。

若要切割，将块切割成所需要的块和空闲块。在所需要的块头部、脚步记录相关信息。在空闲块头部、脚部也记录相关信息。

若不切断，在当前块头部、脚部记录相关信息即可。

**static void \*coalesce(void \*bp)：**



此函数用于合并空闲块。前三种情况已给出。只需要完善最后一种情况。(前后相邻块均为空闲块，三块合为一块)

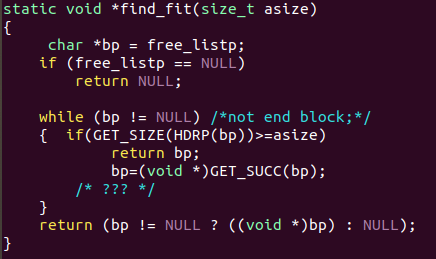
大小等于三块大小相加。然后在前一块的头部录入合并的新块的相关信息，在后一块脚部也录入相关信息。

然后将指针指向新块，即前一块即可。

显式空闲链表：

显示空闲链表与隐式最大的区别就是，专门给出了空闲链表，在寻找空闲块式的效率大大提高。

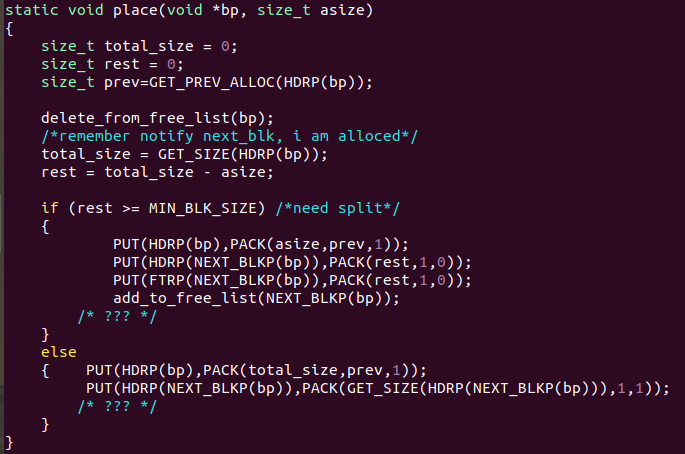
**ﬁnd\_ﬁt()：**



此函数与隐式时同理，用于寻找合适的空闲块。此时只需要从空闲链表里寻找合适的即可。将bp指向空闲链表第一个。如果空闲链表为空（无空闲块）或无合适块返回NULL。

利用GET\_SUCC()找到当前空闲块的后继空闲块。

**place()：**

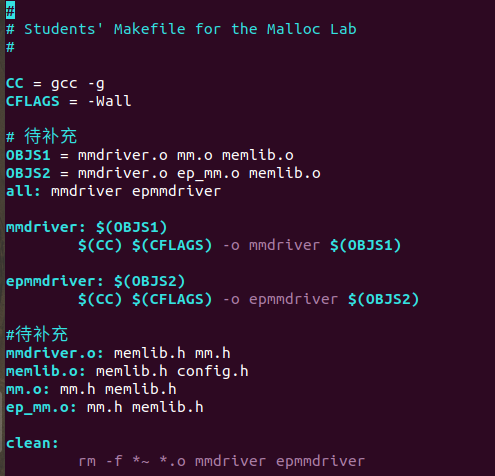


此函数用于切割找到的空闲块。先将找到块从空闲链表删除，然后判断其大小减去所需大小是否大于等于独立空闲块的最小大小。

若需要切割，先在块头部录入相关信息。此时额外需要录入前一相邻块是否被分配，GET\_PREV\_ALLOC()得到即可（分配块无脚部，不需要录入）。然后在切割得到的空闲块头部、脚部录入相关信息，并将其加入空闲链表。

若不需要切割，录入头部，并告知相邻的下一块，前块已被分配。

**Makefile:**



根据相关文件的依赖关系补全makefile。

**编译测试结果：**

