改进方法

计34 董胤蓬 2013011367

1. 页对齐处理

由于有页对齐的情况，所以在申请内存空间的过程中，实际的返回地址不一定为申请的地址，此时需要重新判断申请的此块内存的大小及current_break地址。所以将%ebx作为参数并使用指令int 45后，将返回值%eax赋给%ebx，并重新计算内存块的大小为：%edx = %ebx - old %eax - HEADER_SIZE，这样可以解决页对齐的问题。

2. 多次调用brk解决

采用类似于vector的扩容方式，每次加倍扩容，这样使得扩容次数由O(n)降为O(logn)，可以减少系统调用的次数。

具体的方法为：首先在初始化的时候申请大小为heap_size(100或其他值)的空间，如果当前的内存块大小不够用，则申请heap_size大小的空间，同时heap_size加倍，这样就做到了类似加倍扩容的方式。

3. 内存块的分配

由于采用以上申请内存的方式，所以需要动态地分配内存。当前内存块大小大于所需内存大小时，需要将剩余的部分作为新的内存块以供其他申请利用。具体做法为：当%edx > %ecx时，令%ebx = %eax + %ecx作为下一块内存的地址，并令这块内存的大小为%edx - %ecx，这样可以有效地利用内存。（注：%edx为此块内存大小， %ecx为所需内存大小加上HEADER_SIZE）

4. 相邻的available块合并

实现中，并没有采用在deallocate时使相邻available合并的做法，因为这样做需要记录当前块上一块的地址（因为可能需要同上一块内存合并），有两种方式可以满足，一种为当前块中储存上一块地址，需要额外的空间，另一种为从头开始顺序查

找，浪费时间，均不是很好的方式。实验中采用在分配内存时合并的做法，具体的方法为：

在申请内存块时，如果发现某一内存块为available且大小小于所需大小时，要检查下一内存块状态，如果为available，则将两块内存合并，跳转到loop_begin继续循环，如果为unavailable，则直接跳过，这样在申请时合并内存块，可以提高效率。

5. 内存末尾剩余内存块处理

此处针对以下情况进行优化：如果内存末尾有剩余内存且大小小于所需内存大小，这样在申请新的空间后，需要对末尾的小块内存进行利用。

具体的做法为：在判断某块内存为available且大小不满足需求且为最后一块内存时，在程序中将%ebx指向最后一块地址%eax，而不是current_break，然后加上需要扩容的大小，并进行扩容。这样就不会出现浪费的情况。