OS lab1 document

陆伟嘉
5140219396

1. 八进制打印
   八进制打印与十六进制打印代码接近，只是八进制是 unsigned，用的读取 int 的函数不同。
2. Padding
   Padding 要分有无负号，当有负号时，相当于是让最外层（也就是数字的最低位）执行打空格，而不是之前的让最内层（也就是最高位）打。这样的话，就需要将最外层特殊处理，而由于递归传参的限制，很难判断最外层。所以，便将这个递归写成了一个新函数，让最外层去调新函数，再让新函数自己递归，这样最外层就被特殊化了，就 OK 了。另外，由于这个变动，--width 要变为 width--。
3. %n 的溢出
   此处我一开始是怎么做都不对的，后来发现 putdat 的赋值要在打印 warning 之前完成。由于指针的动态性质，打印 warning 本身也会改变 putdat 导致结果错误。
4. 打印栈信息
   整个过程就是和上学期编译课的 staticlink 的寻址过程是一样的。Ebp 开始往高地址依次存储着旧 ebp，返回地址，参数 1，2..，通过递归追踪。
   注：打印参数要用%08x 才能与参考结果匹配。
5. 通过 cprintf 赋值给返回地址
   由于已经给了 256 位的数组和 cprintf 的限制，便想到让 cprintf 每次只读到想要的值对应的那位结束，也就是将那一位置为\0，但此处要注意由于 little endian，要反着赋值。
6. Overflow
   要跳转到 do_overflow()并不难，只要在 eip 地址获取后将函数地址填进去就可以了，难的是之后如何恢复原来的执行流。一开始我想着把被覆盖掉的地址（原来的跳转地址）放到相应的位置，但感觉想不通，太复杂。后来我想，既然函数都是执行到最后再跳转的，干嘛一定要回到 start_overflow()，直接让它在 overflow_me()的栈上执行就可以了，所以要维持栈，就要把 push ebp, mov esp ebp 跳过，也就是 3 个字节的指令。