**实验五：完成示例4-7的实验，注意观察Relase模式和Debug模式的差异。**

**示例4-7：**

|  |
| --- |
| #include <stdio.h>  int main(int argc, char \*argv[])  {  char str[200];  fgets(str,200,stdin);  printf(str);  return 0;  } |

编译后运行（Release模式）并输入：AAAA%x%x%x%x

我们成功读到了AAAA：AAAA18FE84BB40603041414141（0x41就是ASCII的字母A的值）。

思考：这个41414141是怎么读到的？考虑栈帧状态，参数入栈（字符串str的地址）后，通过%x依次读参数下面的内存数据时，很快就读到了原来函数的局部变量str的数据了。

执行printf(str)语句的时候，对比Debug模式和Relase模式的栈帧结构，如下图4-16所示。



图4-16 对比Debug模式和Relase模式的栈帧结构

Debug模式下，因为开辟了足够大的栈帧并初始化，char str[200]是从靠近EBP的地址分配空间，如果要读到str的地址，需要很多的格式化字符；但是，Relase模式下，可以看到，并没有严格按照制式的栈帧分配，而是考虑运行性能，在执行到printf(str)的时候，栈区自顶到底部分为存着“printf函数参数|fgets函数参数|str数组”的内容，在Main函数的retn语句前，才有一个add esp XX的处理。

如果将AAAA换成地址，第4个%x，换成%s的读取参数指定的地址上的数据呢？是不是就可以读取任意内存地址的数据了？

比如我们输入：AAAA%x%x%x%s

这样就构造了去获取0x41414141地址上的数据的输入。