实验报告

实验名称	实验二：字符串函数的设计

1. 攻击面分析：
   1) 函数 gets_safe ：
      - 输入字符串 str 可能为空，可能导致未定义行为，返回错误-1；
      - 参数 n 为 0；
      - 读取中出现错误（getchar 返回错误）；
      - 输入的字符个数与 n 不符合（超出则需安全地清空多余的缓冲区、小于则需保证 str 指向的字符串符合 C 语言定义）。
   2) 函数 strcpy_safe ：
      - 源字符串为空或目的字符串为空，导致未定义行为；
      - 参数 destsz 等于 0；
      - src 字符串长度与 destsz 不符（均要保证 dest 字符串符合 C 语言对字符串）。
2. 设计思路：
   1) int gets_safe(char* str, rsize_t n)

      流程：

顺序	步骤
①	先判断 str 是否为空，若为空，返回-1
②	再判断 n==0，若成立，str[0] = '\0'置零，返回 0
③	设置 rsize_t 类型变量 len，用于计算输入的数据个数
④	while 循环读取单个字符 c：while(c = getchar() != '\n' && c != EOF) 如果产生读取错误 ferror(stdin)，将 str[0] = '\0'置零，返回-2
⑤	判断 c 是否为回车换行符，则跳出循环，停止读入； 若不是，保存 c 至 str 中，并且 len+1；
⑥	判断 len >= n,若是，则只读取前 n-1 个字符，设置 len = n-1； 并清空缓冲区 while ((c = getchar()) != '\n' && c != EOF); 跳出循环；
⑦	str 末尾处添加空字符 str[len] = '\0';
⑧	返回 len

   2) int strcpy_safe(char* dest, rsize_t destsz, const char* src)

      流程：

顺序	步骤
①	先判断 destsz == 0 或 src == NULL 是否成立，若均不成立，执行第 2 步,否则返回 0；
②	判断目的字符串 dest 是否为空，若是，返回-1，否则执行第 3 步；
③	设置 rsize_t 类型变量 n，用于作为 dest 复制时的下标； 并计算 src 长度 len_of_src;
④	比较 src 长度与 destsz 的大小：若 len_of_src >= destsz 执行第 5 步；

	若小于，执行第 6 步；
⑤	循环复制前 (destsz-1) 个字符：while(n < destsz-1)，使用下标方式复制：dest[n] = src[n]，每次复制后 n++； 循环结束，目的缓冲区最后一个字符置零； 返回 destsz – 1。
⑥	循环复制前整个 src 字符串：while (n <len_of_src) ,使用下标方式复制：dest[n] = src[n]，每次复制后 n++； 循环结束，目的缓冲区最后一个字符置零； 返回 len_of_src；

3) 测试用例及结果：

- int gets_safe(char* str, rsize_t n)测试用例：

① stdin: “123456”, “1234567”, “12345678”, “\n”
结果如图 1。

② char* buf = 0;
gets_safe(buf, sizeof(buf));
结果如图 2。

③ char* buf[8];
gets_safe(buf, 0);
结果如图 2。

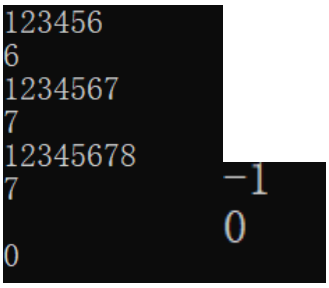


图 1、图 2

- int strcpy_safe(char* dest, rsize_t destsz, const char* src)测试用例：

① char dest[8];
char *src = “12345”;
strcpy_safe(dest, sizeof(dest), src);

② char dest[8];
char *src = “1234567”;
strcpy_safe(dest, sizeof(dest), src);

③ char dest[8];
char *src = “12345678”;
strcpy_safe(dest, sizeof(dest), src);

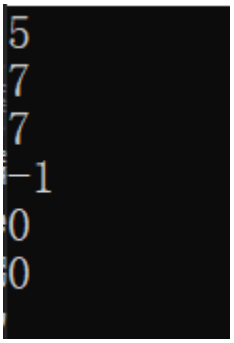
④ char *src = “1234567”;
strcpy_safe(0, 8, src);

⑤ char dest[8];
strcpy_safe(dest, sizeof(dest), 0);

⑥ char dest[8];
char *src = “12345678”;

strcpy_safe(dest, 0, src);

结果图：



3. 心得体会

本次实验，我了解到了如何安全地清空缓冲区、使用 ferror 函数测试读取错误、stdin 输入操作的机制以及学习了 MSDN 上微软提供的 strsafe 函数。

在实验过程中，因为基础不够牢靠，对于许多知识点并不熟悉，这告诉我以后应当夯实基础，踏实学习理论，在遇到问题时不能只靠浏览搜索引擎上的博客知识，应当仔细搞清楚其中原理方能有所进步。