| 1 void CWE416\_Use\_After\_Free\_\_malloc\_free\_char\_01\_bad() 2 { 3 char \* data; 4 /\* Initialize data \*/ 5 data = NULL; 6 data = (char \*)malloc(100\*sizeof(char)); 7 **if** (data == NULL) {exit(-1);} 8 memset(data, 'A', 100-1); 9 data[100-1] = '\0'; 10 /\* POTENTIAL FLAW: Free data in the source - the bad sink attempts to use data \*/  11 free(data); 12 /\* POTENTIAL FLAW: Use of data that may have been freed \*/ 13 printLine(data); 14 /\* POTENTIAL INCIDENTAL - Possible memory leak here if data was not freed \*/ 15 } |
| --- |

From the above code, we can see a use-after-free caused by the statement at line 13. Specifically, the code declares a pointer “data” at line 3, and makes “data” point to a buffer allocated on the heap at line 6. The buffer is initialized at line 8 and line 9 and then freed at line 11. However, the buffer, after being freed, is used again at line 13 (the code of function “printIntLine” is included in “Include/io.c”). Thus, a use-after-free happens.