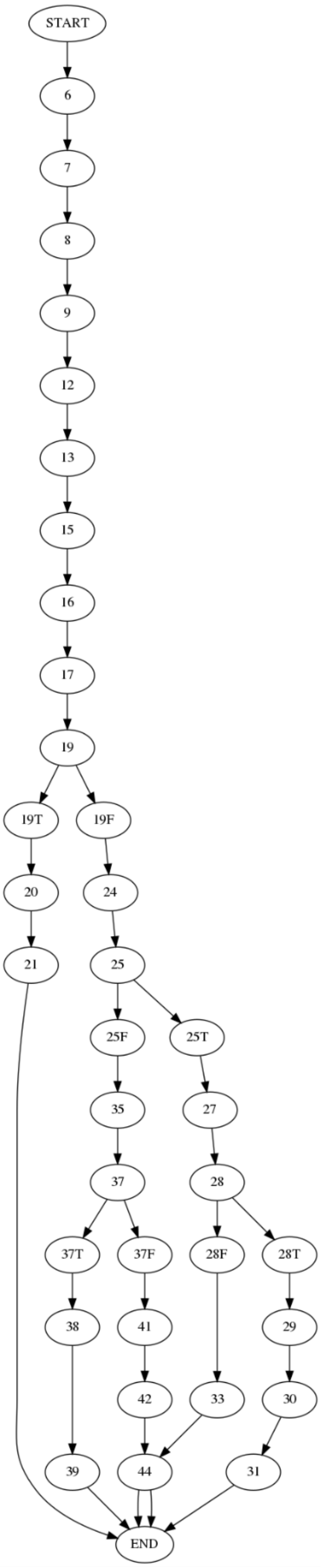
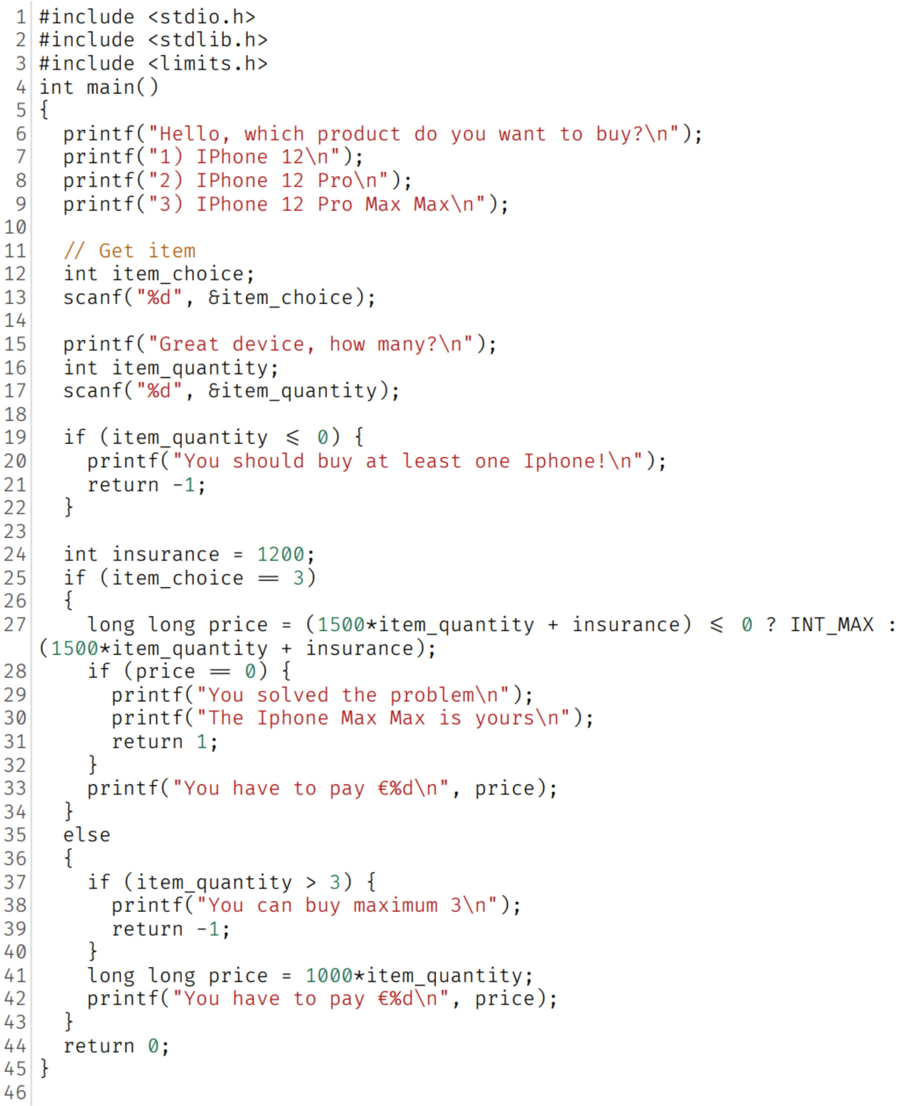
Taint analysis exercises

Initial premise: both exercises have been analyzed with state numbers corresponding with their line numbers. Additionally, both have been fixed in-line without changing the line number, so CFGs for both do not differ between tainted and fixed versions. CFGs were made using DOT and exported in PNG. Finally, the all curly brackets have been left out of the tables for reading clarity (due to a large amount of clutter).



Exercise 1

The CFG for the integer overflow can be found on the left. The two tables, the first for the initial iteration comprising kill, get, out and in = {}, and the second comprising the second iteration (no more are needed), can be found in the following pages.



```
 1 #include <stdio.h>
 2 #include <stdlib.h>
 3 #include <limits.h>
 4 int main()
 5 {
 6   printf("Hello, which product do you want to buy?\n");
 7   printf("1) IPhone 12\n");
 8   printf("2) IPhone 12 Pro\n");
 9   printf("3) IPhone 12 Pro Max Max\n");
10
11   // Get item
12   int item_choice;
13   scanf("%d", &item_choice);
14
15   printf("Great device, how many?\n");
16   int item_quantity;
17   scanf("%d", &item_quantity);
18
19   if (item_quantity <= 0) {
20     printf("You should buy at least one Iphone!\n");
21     return -1;
22   }
23
24   int insurance = 1200;
25   if (item_choice == 3)
26   {
27     long long price = (1500*item_quantity + insurance) <= 0 ? INT_MAX :
(1500*item_quantity + insurance);
28     if (price == 0) {
29       printf("You solved the problem\n");
30       printf("The Iphone Max Max is yours\n");
31       return 1;
32     }
33     printf("You have to pay €%d\n", price);
34   }
35   else
36   {
37     if (item_quantity > 3) {
38       printf("You can buy maximum 3\n");
39       return -1;
40     }
41     long long price = 1000*item_quantity;
42     printf("You have to pay €%d\n", price);
43   }
44   return 0;
45 }
46
```

A little note has to be made regarding the taint analysis itself for the vulnerable code. The if on line 37, checking whether the item_quantity variable in that case is greater than 3, automatically sanitizes it. This is because of the previous “if” on line 19. Therefore, the item_quantity variable in this case falls in the range [1, 2]. This, however, does not hold for the item_choice == 3 case.

In order to address this issue, the code was fixed and specifically in line 27. The code written above features two methods of fixing the issue, which are not reliant on each other and can be used separately.

The first way is adding an if check, implemented with a C ternary in order to check whether the price did overflow or reach 0. Doing this sets the price to the INT_MAX variable.

The second way, which is more elegant and should be preferred, is setting the type of the price variable to long long. This mathematically assures that it will never overflow: since item_quantity may reach 2^{16} -1 as its max value, multiplying it by 1500 (2^{10}) makes it impossible to even reach 2^{64}-1. This implementation, however, is not platform-independent as the C type system does not enforce upper bounds for types – this means that both int and long long could theoretically be of length 128 bits. In order to be fully protected, the __builtin_mul_overflow function should be used.

The following table features the aforementioned iterations for the tainted version of the code.

	gen	kill	in	out
6				
7				
8				
9				
12				
13	item_choice			item_choice
15				
16				
17	item_quantity			item_quantity
19				
19T				
20				
21				
19F				
24				
25				
25T				
27	[item_quantity->T | insurance -> T]price	[item_quantity->F & insurance -> F]price		price
28				
28T				
29				
30				
31				
28F				
33				
25F				
35				
37				
37T				
38				
39				
37F		item_quantity		

41				
42				
44				
It. 2				
	gen	kill	in	out
6				
7				
8				
9				
12				
13	item_choice			item_choice
15			item_choice	item_choice
16			item_choice	item_choice
17	item_quantity		item_choice	item_choice, item_quantity
19			item_choice, item_quantity	item_choice, item_quantity
19T			item_choice, item_quantity	item_choice, item_quantity
20			item_choice, item_quantity	item_choice, item_quantity
21			item_choice, item_quantity	item_choice, item_quantity
19F			item_choice, item_quantity	item_choice, item_quantity
24			item_choice, item_quantity	item_choice, item_quantity
25			item_choice, item_quantity	item_choice, item_quantity
25T			item_choice, item_quantity	item_choice, item_quantity
27	[item_quantity->T | insurance -> T]price	[item_quantity->F & insurance -> F]price	item_choice, item_quantity	item_choice, item_quantity, price
28			item_choice, item_quantity, price	item_choice, item_quantity, price
28T			item_choice, item_quantity, price	item_choice, item_quantity, price
29			item_choice, item_quantity, price	item_choice, item_quantity, price
30			item_choice, item_quantity, price	item_choice, item_quantity, price
31			item_choice, item_quantity, price	item_choice, item_quantity, price
28F			item_choice, item_quantity, price	item_choice, item_quantity, price
33			item_choice, item_quantity, price	item_choice, item_quantity, price
25F			item_choice, item_quantity	item_choice, item_quantity

35			item_choice, item_quantity	item_choice, item_quantity
37			item_choice, item_quantity	item_choice, item_quantity
37T			item_choice, item_quantity	item_choice, item_quantity
38			item_choice, item_quantity	item_choice, item_quantity
39			item_choice, item_quantity	item_choice, item_quantity
37F		item_quantity	item_choice, item_quantity	item_choice
41			item_choice	item_choice
42			item_choice	item_choice
44			item_choice	item_choice

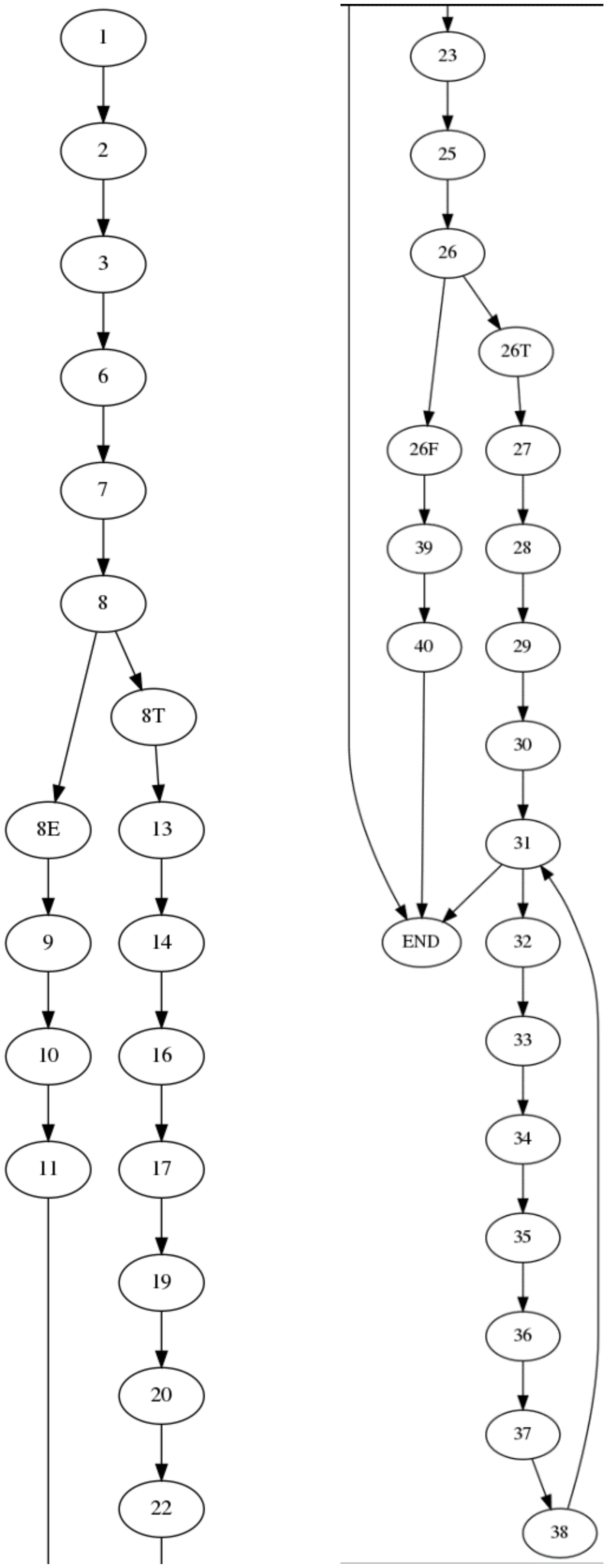
The following tables, on the other hand, show the taint analysis of the fixed code.

	gen	kill	in	out
6				
7				
8				
9				
12				
13	item_choice			item_choice
15				
16				
17	item_quantity			item_quantity
19				
19T				
20				
21				
19F				
24				
25				
25T				
27	price	price		
28				
28T				
29				
30				
31				
28F				
33				
25F				
35				
37				
37T				
38				

39				
37F		item_quantity		
41				
42				
44				
It. 2				
	gen	kill	in	out
6				
7				
8				
9				
12				
13	item_choice			item_choice
15			item_choice	item_choice
16			item_choice	item_choice
17	item_quantity		item_choice	item_choice, item_quantity
19			item_choice, item_quantity	item_choice, item_quantity
19T			item_choice, item_quantity	item_choice, item_quantity
20			item_choice, item_quantity	item_choice, item_quantity
21			item_choice, item_quantity	item_choice, item_quantity
19F			item_choice, item_quantity	item_choice, item_quantity
24			item_choice, item_quantity	item_choice, item_quantity
25			item_choice, item_quantity	item_choice, item_quantity
25T			item_choice, item_quantity	item_choice, item_quantity
27	price	price	item_choice, item_quantity	item_choice, item_quantity
28			item_choice, item_quantity	item_choice, item_quantity
28T			item_choice, item_quantity	item_choice, item_quantity
29			item_choice, item_quantity	item_choice, item_quantity
30			item_choice, item_quantity	item_choice, item_quantity
31			item_choice, item_quantity	item_choice, item_quantity
28F			item_choice, item_quantity	item_choice, item_quantity
33			item_choice, item_quantity	item_choice, item_quantity

25F			item_choice, item_quantity	item_choice, item_quantity
35			item_choice, item_quantity	item_choice, item_quantity
37			item_choice, item_quantity	item_choice, item_quantity
37T			item_choice, item_quantity	item_choice, item_quantity
38			item_choice, item_quantity	item_choice, item_quantity
39			item_choice, item_quantity	item_choice, item_quantity
37F		item_quantity	item_choice, item_quantity	item_choice
41			item_choice	item_choice
42			item_choice	item_choice
44			item_choice	item_choice

Exercise 2

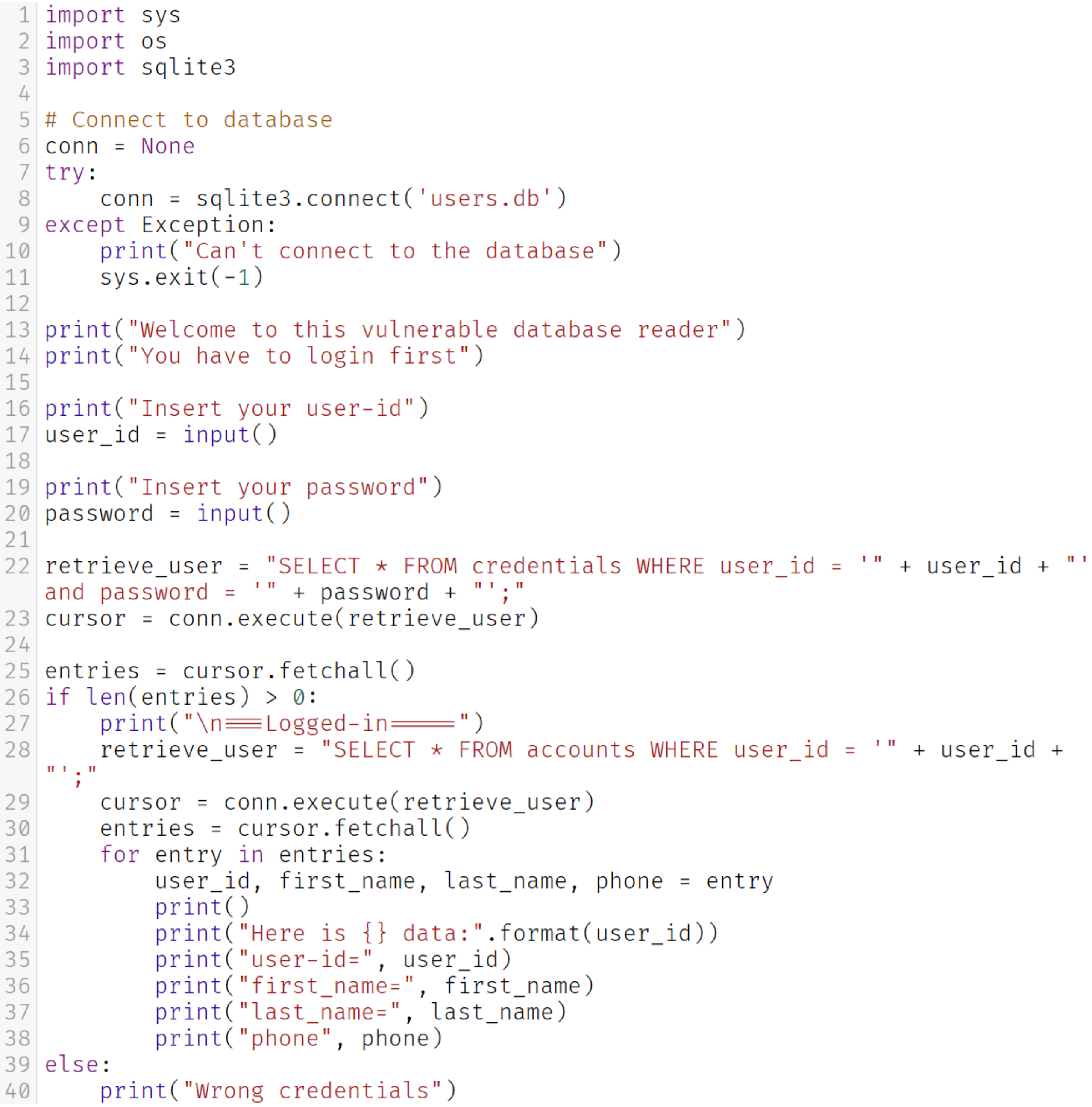


The CFG for the SQL injection exercise can be found on the left, split in two parts for clarity. The code can be found in the following page.

The main reasoning points about the exercise are the following:

- Prepared statements fix lines 22/23 and 28/29. The user_id and password variables do remain tainted, but are sanitized in the process of addition to the statement so that the cursor is not tainted at all
- The entries variable, at a first glance, may appear tainted if and only if the cursor was tainted by a malicious statement. However, we can never be sure of what we are retrieving from a DB (maybe some other malicious code was injected by a different program), so we consider it tainted no matter what. Therefore, we need to properly sanitize with a map the variables unpacked in the loop starting from line 31. This ensures that echoed variable are not tainted. Due to this, in the analysis the entries variables is shown as always tainted instead of adding a condition based on cursor ([cursor=T]entries for gen and [cursor=F]entries for kill).
- An additional point, not included in the main taint analysis (because it would have generated confusion), is that libraries may not be as safe as they appear to. For example, the sqlite3 library may contain a vulnerability in a certain version, and this may compromise any program written with it. Therefore, it may be considered as tainted. For the purposes of this exercise, it hasn't been.

As before, the fixes have been inserted as one liners, therefore leaving the line numbering schema intact.



```python
import sys
import os
import sqlite3

# Connect to database
conn = None
try:
    conn = sqlite3.connect('users.db')
except Exception:
    print("Can't connect to the database")
    sys.exit(-1)

print("Welcome to this vulnerable database reader")
print("You have to login first")

print("Insert your user-id")
user_id = input()

print("Insert your password")
password = input()

retrieve_user = "SELECT * FROM credentials WHERE user_id = '" + user_id + "' and password = '" + password + "';"
cursor = conn.execute(retrieve_user)

entries = cursor.fetchall()
if len(entries) > 0:
    print("\n===Logged-in=====")
    retrieve_user = "SELECT * FROM accounts WHERE user_id = '" + user_id + "';"
    cursor = conn.execute(retrieve_user)
    entries = cursor.fetchall()
    for entry in entries:
        user_id, first_name, last_name, phone = entry
        print()
        print("Here is {} data:".format(user_id))
        print("user-id=", user_id)
        print("first_name=", first_name)
        print("last_name=", last_name)
        print("phone", phone)
else:
    print("Wrong credentials")
```

In the following pages we can find the tables for the taint analysis. There are six tables, which represent:

- Tainted code analysis – first step
- Tainted code analysis – second step (two cell differences due to a loop)
- Tainted code analysis – final table
- Untainted code analysis – first step
- Untainted code analysis – second step
- Untainted code analysis – final table (just a cell difference due to a loop)

	SQL injection first			
	gen	kill	in	out
1				
2				
3				
6				
7				
8				
8E				
9				
10				
11				
8T				
13				
14				
16				
17	user_id			user_id
19				
20	password			password
22	[user_id = T | password = T]retrieve_user	[user_id = F & password = F]retrieve_user		retrieve_user
23	[retrieve_user = T]cursor	[retrieve_user = F]cursor		cursor
25	entries			entries
26				
26F				
39				
40				
26T				
27				

28	[user_id = T]retrieve_user	[user_id = F]retrieve_user		retrieve_user
29	[retrieve_user = T]cursor	[retrieve_user = F]cursor		cursor
30	entries			entries
31	entry			entry
32	[entry = T]user_id, [entry = T]first_name, [entry = T]last_name, [entry = T]phone	[entry = F]user_id, [entry = F]first_name, [entry = F]last_name, [entry = F]phone		user_id, first_name, last_name, phone
33				
34				
35				
36				
37				
38				
	SQL injection second			
	gen	kill	in	out
1				
2				
3				
6				
7				
8				
8E				
9				
10				
11				
8T				
13				
14				
16				

17	user_id			user_id
19			user_id	user_id
20	password		user_id	user_id, password
22	[user_id = T | password = T]retrieve_user	[user_id = F & password = F]retrieve_user	user_id, password	user_id, password, retrieve_user
23	[retrieve_user = T]cursor	[retrieve_user = F]cursor	user_id, password, retrieve_user	user_id, password, retrieve_user, cursor
25	entries		user_id, password, retrieve_user, cursor	user_id, password, retrieve_user, cursor, entries
26			user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
26F			user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
39			user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
40			user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
26T			user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
27			user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
28	[user_id = T]retrieve_user	[user_id = F]retrieve_user	user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
29	[retrieve_user = T]cursor	[retrieve_user = F]cursor	user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
30	entries		user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
31	entry		user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries, entry
32	[entry = T]user_id, [entry = T]first_name, [entry = T]last_name, [entry = T]phone	[entry = F]user_id, [entry = F]first_name, [entry = F]last_name, [entry = F]phone	user_id, password, retrieve_user, cursor, entries, entry	user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone
33			user_id, password, retrieve_user, cursor, entries, entry, user_id,	user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone

			first_name, last_name, phone	
34			user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone	user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone
35			user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone	user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone
36			user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone	user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone
37			user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone	user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone
38			user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone	user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone
	SQL injection third			
	gen	kill	in	out
1				
2				
3				
6				
7				
8				
8E				
9				
10				

11				
8T				
13				
14				
16				
17	user_id			user_id
19			user_id	user_id
20	password		user_id	user_id, password
22	[user_id = T | password = T]retrieve_user	[user_id = F & password = F]retrieve_user	user_id, password	user_id, password, retrieve_user
23	[retrieve_user = T]cursor	[retrieve_user = F]cursor	user_id, password, retrieve_user	user_id, password, retrieve_user, cursor
25	entries		user_id, password, retrieve_user, cursor	user_id, password, retrieve_user, cursor, entries
26			user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
26F			user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
39			user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
40			user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
26T			user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
27			user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
28	[user_id = T]retrieve_user	[user_id = F]retrieve_user	user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
29	[retrieve_user = T]cursor	[retrieve_user = F]cursor	user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
30	entries		user_id, password, retrieve_user, cursor, entries	user_id, password, retrieve_user, cursor, entries
31	entry		user_id, password, retrieve_user, cursor, entries, entry, user_id,	user_id, password, retrieve_user, cursor, entries, entry

			first_name, last_name, phone	
32	[entry = T]user_id, [entry = T]first_name, [entry = T]last_name, [entry = T]phone	[entry = F]user_id, [entry = F]first_name, [entry = F]last_name, [entry = F]phone	user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone	user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone
33			user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone	user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone
34			user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone	user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone
35			user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone	user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone
36			user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone	user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone
37			user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone	user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone
38			user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone	user_id, password, retrieve_user, cursor, entries, entry, user_id, first_name, last_name, phone

SQL injection fixed first

	gen	kill	in	out
1				
2				

3				
6				
7				
8				
8E				
9				
10				
11				
8T				
13				
14				
16				
17	user_id			user_id
19				
20	password			password
22				
23				
25	entries			entries
26				
26F				
39				
40				
26T				
27				
28				
29				
30	entries			entries
31	[entries = T]entry	[entries = F]entry		entry
32				

33				
34				
35				
36				
37				
38				
	SQL Injection fixed second			
	gen	kill	in	out
1				
2				
3				
6				
7				
8				
8E				
9				
10				
11				
8T				
13				
14				
16				
17	user_id			user_id
19			user_id	user_id
20	password		user_id	user_id, password
22			user_id, password	user_id, password
23			user_id, password	user_id, password
25	entries		user_id, password	user_id, password, entries

26			user_id, password, entries	user_id, password, entries
26F			user_id, password, entries	user_id, password, entries
39			user_id, password, entries	user_id, password, entries
40			user_id, password, entries	user_id, password, entries
26T			user_id, password, entries	user_id, password, entries
27			user_id, password, entries	user_id, password, entries
28			user_id, password, entries	user_id, password, entries
29			user_id, password, entries	user_id, password, entries
30	entries		user_id, password, entries	user_id, password, entries
31	[entries = T]entry	[entries = F]entry	user_id, password, entries	user_id, password, entries, entry
32			user_id, password, entries, entry	user_id, password, entries, entry
33			user_id, password, entries, entry	user_id, password, entries, entry
34			user_id, password, entries, entry	user_id, password, entries, entry
35			user_id, password, entries, entry	user_id, password, entries, entry
36			user_id, password, entries, entry	user_id, password, entries, entry
37			user_id, password, entries, entry	user_id, password, entries, entry
38			user_id, password, entries, entry	user_id, password, entries, entry
	SQL Injection fixed third			
	gen	kill	in	out
1				
2				
3				
6				
7				
8				
8E				
9				

10				
11				
8T				
13				
14				
16				
17	user_id			user_id
19			user_id	user_id
20	password		user_id	user_id, password
22			user_id, password	user_id, password
23			user_id, password	user_id, password
25	entries		user_id, password	user_id, password, entries
26			user_id, password, entries	user_id, password, entries
26F			user_id, password, entries	user_id, password, entries
39			user_id, password, entries	user_id, password, entries
40			user_id, password, entries	user_id, password, entries
26T			user_id, password, entries	user_id, password, entries
27			user_id, password, entries	user_id, password, entries
28			user_id, password, entries	user_id, password, entries
29			user_id, password, entries	user_id, password, entries
30	entries		user_id, password, entries	user_id, password, entries
31	[entries = T]entry	[entries = F]entry	user_id, password, entries, entry	user_id, password, entries, entry
32			user_id, password, entries, entry	user_id, password, entries, entry
33			user_id, password, entries, entry	user_id, password, entries, entry
34			user_id, password, entries, entry	user_id, password, entries, entry
35			user_id, password, entries, entry	user_id, password, entries, entry
36			user_id, password, entries, entry	user_id, password, entries, entry

37			user_id, password, entries, entry	user_id, password, entries, entry
38			user_id, password, entries, entry	user_id, password, entries, entry