Buggy TTY-Solitaire

IMPORTANT INFORMATION

The version of Solitaire used was from https://github.com/mpereira/tty-solitaire version 1.1.1

Assignment is provided with questions and solution given after

BUG 1

Original Buggy Code for Bug 1:
In src/game.c:

Change lines 138 and 139 by removing the '=' in the loop conditions

```
138   for (int i = ACE; i < KING; i++) {
139     for (int j = DIAMONDS; j < CLUBS; j++) {
```

Bug 1

Customers calling in are saying that the game is completely useless in that they can't get the game started. When they run the program as in

```
./ttysolitaire-gdb
```

and when they try to start the game, it crashes.

NOTE: Bug 1 must be fixed first before you can move on.

You are to file a bug report for this issue. You should definitely try to recreate the bug to verify. After verifying the bug, you should type in the report below.

The fields you will be required to provide are as follows (for a complete description, see the "Filing a Bug" page in the tutorials):

- Product or program **name** for which you are filing a bug
- A one-line **summary** that uniquely identifies the issue
- **Status** (New, Unconfirmed, etc.)
- **Severity**
- **Priority** (while priority is normally set by management, give this one your best guess)
- **Comments** (the comments are more free form, but you should still provide the comments as though you are doing this for work and be professional)

Now, using GDB or DDD (or ANY debugger of your choice), find the defect and fix it.

What line(s) of code was/were problematic? List the line(s) that contained the defect and what function it was in, and then list the change necessary to fix the defect.

Please provide your logbook entry for this bug (including all the necessary pieces we discussed in class up to now, including application of the Scientific Method).

PLEASE KEEP IN MIND that it is expected that this puzzle may take a bit longer, so your logbook entries should be a bit more extensive!

SOLUTION TO BUG 1

In src/game.c:

lines 138 and 139 SHOULD have used <= for the loop conditions. As in:

```
138    for (int i = ACE; i <= KING; i++) {
139      for (int j = DIAMONDS; j <= CLUBS; j++) {
```

Removing the '=' causes the bug for the assignment

But KING and CLUBS are legitimate parts.

BUG 2

Original Buggy Code for Bug 2:

Within the print_reverse_stack function of stack.c, change the top of the function so that it alters the stack variable that was passed in. As in:

```
void print_reverse_stack(struct stack *stack) {
     struct stack *last_card = stack;
     struct stack *next_card = stack->next;
     //reverse the stack
     …
```

Bug 2

Our first customer that got the fix for Bug 1 already called in another bug. Here is what they said:

"When I move the cursor around, it just looks weird... and the game looks weird. Also, I can't always flip cards over that I should be able to."

NOTE: You can move on to bug 3 or 4 first if you wish. That is, bugs 3 and 4 do not depend on this bug

You are to file a bug report for this issue (**it is OK to file separate bugs below in separate enumerated lists if you feel this is needed**). You should definitely try to recreate the bug to verify. After verifying the bug, you should type in the report below.

The fields you will be required to provide are as follows (for a complete description, see the "Filing a Bug" page in the tutorials):

- Product or program **name** for which you are filing a bug
- A one-line **summary** that uniquely identifies the issue
- **Status** (New, Unconfirmed, etc.)
- **Severity**
- **Priority** (while priority is normally set by management, give this one your best guess)
- **Comments** (the comments are more free form, but you should still provide the comments as though you are doing this for work and be professional)

Now, using GDB or DDD (or ANY debugger of your choice), find the defect and fix it.

What line(s) of code was/were problematic? List the line(s) that contained the defect and what function it was in, and then list the change necessary to fix the defect.

Please provide your logbook entry for this bug (including all the necessary pieces we discussed in class up to now, including application of the Scientific Method).

PLEASE KEEP IN MIND that it is expected that this puzzle may take a bit longer, so your logbook entries should be a bit more extensive!

SOLUTION TO BUG 2

To fix, here are methods that duplicate the stack already, or at least make a copy of the stack that can be altered however desired. There are several fixes you can do here, any are acceptable.

1) The easiest fix is to change the top two lines

```
struct stack *last_card = stack_dup(stack);
struct stack *next_card = last_card->next;
```

2) You can comment out the entire while loop and set last_card to the stack_reverse call instead as in:

```
struct stack *last_card = stack_reverse(stack);
struct stack *next_card = last_card->next;

//remove while loop
```

BUG 3

Original Buggy Code for Bug 3:
Insert a bug in keyboard.c, in keyboard_event function for the case KEY_SPACEBAR,

```
246       struct card *tmp;
247       if (stack_pop(&(deck->stock))) {
247            tmp = stack_pop(&(deck->stock));
```

Bug 3

Another customer with Bug 1 fixed called in and said:

"When I am drawing from the stock pile, there is no way I am getting all the cards!"

(draw through the stock to figure this one out)

NOTE: You do NOT need to fix Bug 2 first before working on this bug. You can also work on Bug 4 without having fixed this bug.

You are to file a bug report for this issue. You should definitely try to recreate the bug to verify. After verifying the bug, you should type in the report below.

The fields you will be required to provide are as follows (for a complete description, see the "Filing a Bug" page in the tutorials):

- Product or program **name** for which you are filing a bug
- A one-line **summary** that uniquely identifies the issue
- **Status** (New, Unconfirmed, etc.)
- **Severity**
- **Priority** (while priority is normally set by management, give this one your best guess)
- **Comments** (the comments are more free form, but you should still provide the comments as though you are doing this for work and be professional)

Now, using GDB or DDD (or ANY debugger of your choice), find the defect and fix it.

What line(s) of code was/were problematic? List the line(s) that contained the defect and what function it was in, and then list the change necessary to fix the defect.

Please provide your logbook entry for this bug (including all the necessary pieces we discussed in class up to now, including application of the Scientific Method).

PLEASE KEEP IN MIND that it is expected that this puzzle may take a bit longer, so your logbook entries should be a bit more extensive!

SOLUTION FOR BUG 3

the problem is in lines 247-248 (or there about, if they fixed other bugs first, these line numbers might change). The if statement pops off the stack, and immediately after that, it sets tmp to another stack pop.

The lines should be changed to:

```
246          struct card *tmp = stack_pop(&(deck->stock));
247            if (tmp) {
248              //tmp = stack_pop(&(deck->stock));
```

i.e., stack_pop should be called only once when this is done

BUG 4

Original Buggy Code for Bug 4:

Edit keyboard.c if statement starting at line 167 such that when card is added to the Foundation stack for the first time, it is added with a pointer to itself, causing an infinite loop. As in:

```
167  if (foundation_stack(*destination)) {
168       //make sure foundation card has a dummy next value
169       (*destination)->next = *destination;
170  }
```

Bug 4

We received the following comment from another customer:

"Any time I move a card from the upper right 'foundation' section back down to the playing area, the game freezes."

NOTE: You can recreate this bug without having fixed bugs 2 or 3.

You are to file a bug report for this issue. You should definitely try to recreate the bug to verify. After verifying the bug, you should type in the report below.

The fields you will be required to provide are as follows (for a complete description, see the "Filing a Bug" page in the tutorials):

- Product or program **name** for which you are filing a bug
- A one-line **summary** that uniquely identifies the issue
- **Status** (New, Unconfirmed, etc.)
- **Severity**
- **Priority** (while priority is normally set by management, give this one your best guess)
- **Comments** (the comments are more free form, but you should still provide the comments as though you are doing this for work and be professional)

Now, using GDB or DDD (or ANY debugger of your choice), find the defect and fix it.

What line(s) of code was/were problematic? List the line(s) that contained the defect and what function it was in, and then list the change necessary to fix the defect.

Please provide your logbook entry for this bug (including all the necessary pieces we discussed in class up to now, including application of the Scientific Method).

PLEASE KEEP IN MIND that it is expected that this puzzle may take a bit longer, so your logbook entries should be a bit more extensive!

SOLUTION FOR BUG 4

It should be changed so that (according to the comment above) it points to a dummy card. Any variations of the following is acceptable:

```
169                 struct stack **dummy;
170                 stack_malloc(dummy);
171                 stack_init(*dummy);
172                 (*destination)->next = *dummy;
```