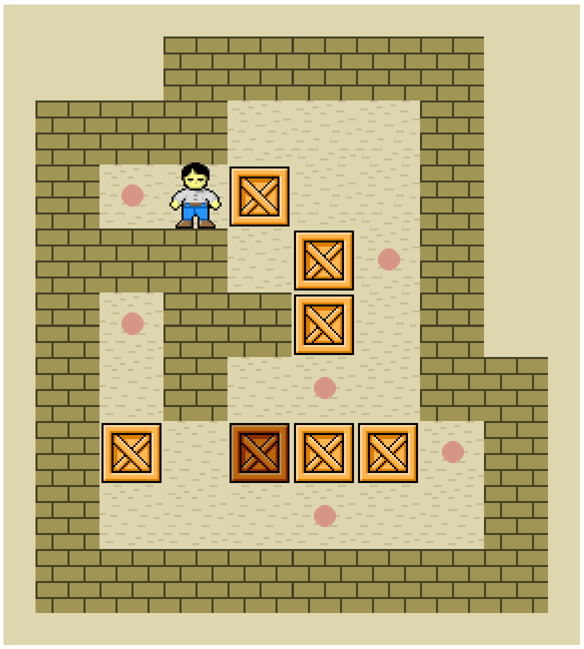
12.1 ZyProject1: Sokoban

This lab will be available until November 19th, 11:59 PM PST



Lab Exam Instructions

This is an individual assignment due on **Friday 11/18 at 11:59 pm.** Any submissions made after Friday, up to Saturday at 11:59PM, will receive a 20% score deduction. As an open book project, you may reference notes and other course material, but you may not work together, or receive help from anyone except staff members. Otherwise, it is just like a normal lab assignment: you may use PythonTutor or other IDEs, and you may submit as many times as you'd like. We will not be style-checking this assignment.

MOSS: A Word on Plagiarism Detection

Please keep in mind that as computer science instructors, we care about academic dishonesty just as strongly as our peers in other fields of study. Unlike our peers, however, our field of expertise and the nature of the material allows us to create tools that enhance our detection of plagiarism. I'm sure you have heard of similar tools, such as TurnItIn.com, and other plagiarism detection systems for essays. Since our field is more mathematical, however, we are capable of considering not only the actual text written on the screen but the overall flow and intent of the program, and mathematically compare it to that of another to determine the level of similarity between the two. Therefore, I highly suggest that you do not "collaborate" with anyone else, as we will be able to detect it. If we catch you cheating, you will receive an automatic zero in the class, and your case will be sent to the Office of Academic Integrity & Student Conduct. With that being said, let's all take a deep breath and try to have some fun!

Preamble

At this point in the course, you have enough knowledge about Python to do more than you think. Many simple games, for example, can be written with what we've already covered. So let's do it! Let's create a game of Sokoban!

Concepts being tested

- Data Structures
  - Nested Lists
  - Potentially Dictionaries
- Branching
- Loops
  - For/While loops
  - Nested Loops
- Critical thinking, planning, & organization

The Game

If you're not familiar with the game, try your hand at it here (spend a few minutes learning the rules of the game). The object of the game is to move all of the boxes onto a target.

So we're all on the same page, let's establish how the game works. The game has 4 controls:

- w - move the sprite up
- s - move the sprite down
- a - move the sprite left
- d - move the sprite right
- q - quit the game
- space - reset the board

These controls are also outlined in the provided game_settings.py file.

The sprite (i) can move in empty spaces of the board, but cannot pass through walls (+). It can push a single box (!) into an empty or target (o) space, but cannot push more than one at a time. In other words, boxes cannot be pushed into other boxes or walls. Pushing the box right:

```
+ + + + + + + +
+   .         +
+   i !   o   +
+       !   o +
+             +
+ + + + + + + +
```

results in this board:

```
+ + + + + + + +
+   .         +
+     i ! o   +
+       !   o +
+             +
+ + + + + + + +
```

When the box is pushed onto a target, it is converted from a non-statisfied box (!) into a satisfied box (.). Moving the sprite right once more, does exactly that:

```
+ + + + + + + +
+   .         +
+       i .   +
+       !   o +
+             +
+ + + + + + + +
```

Note that satisfied boxes (.) can be moved off the target and convert back into non-statisfied boxes (!). Additionally, when the sprite stands on a target space, its string is changed to indicate a target is below (I):

```
+ + + + + + + +
+   .         +
+         I ! +
+       !   o +
+             +
+ + + + + + + +
```

Once the sprite moves off the target, both spaces revert back (I to i and o):

```
+ + + + + + + +
+   .     i   +
+         o ! +
+       !   o +
+                       +
+  +  +  +  +  +  +  +
```

The game ends when a win is detected or the user quits the program

- Quitting prints the message: "Goodbye"
- Winning prints the message: "You Win!"

General Structure of the Program

When boiled down, most simple games have the same repeated set of actions:

- the program provides information about the current game state to the player
- the user takes action through input
- the program reacts
  - the program identifies the input
  - the program updates the game state
- repeat forever, or until the user wins or loses

This *loop* is often referred to as the game loop, and is where the bulk of the game's code lives. Generally, there are things the program must do before the game starts and after it ends:

- Preparation; initialize counter variables and copies of important information
- Game Loop
- Exit Protocol

main.py

This file is where you will write Sokoban.

game_settings.py

This is a read-only file that provides global variables for the different space types (sprite, empty space, box, etc.), valid user controls, and the game board. To pass all test cases, ensure you use these variables throughout your code, as some test cases will change their values and will expect your code to adapt.

Suggestions

Things to consider before developing your program:

- How can you easily find or track the sprite location to avoid repetitive searches?
- Can you do the same for tracking the win condition?
- Can you generalize your logic so that regardless of the direction of the movement, the code is the same?
- What do you need to reset the game?

Sample Game:

```
+ + + + + + + +
+   .         +
+ i   !   o   +
+       !   o +
+             +
+ + + + + + + +

d (Player moves right)
+ + + + + + + +
+   .         +
+   i !   o   +
+       !   o +
+             +
+ + + + + + + +

d
+ + + + + + + +
+   .         +
+     i ! o   +
+       !   o +
+             +
+ + + + + + + +

d (Player pushes the box onto the target, converting the BOX_NS
into BOX_S)
+ + + + + + + +
+   .         +
+       i .   +
+       !   o +
+             +
+ + + + + + + +

d (Player pushes the box off the target, converting BOX_S to
BOX_NS, and steps onto the target, converting SPRITE into SPRITE_T)
+ + + + + + + +
+   .         +
+         I ! +
+       !   o +
+             +
+ + + + + + + +
  (Player resets the game)
+ + + + + + + +
+   .         +
+ i   !   o   +
+       !   o +
+             +
+ + + + + + + +

d
+ + + + + + + +
+   .         +
+   i !   o   +
+       !   o +
+             +
+ + + + + + + +

d
+ + + + + + + +
+   .         +
+     i ! o   +
+       !   o +
+             +
+ + + + + + + +

w
+ + + + + + + +
+   . i       +
+       ! o   +
+       !   o +
+             +
+ + + + + + + +

d
+ + + + + + + +
+   .   i     +
+       ! o   +
+       !   o +
+             +
+ + + + + + + +

s (Player attempts to push more than one box)
+ + + + + + + +
+   .   i     +
+       ! o   +
+       !   o +
+             +
+ + + + + + + +

a
+ + + + + + + +
+   . i       +
+       ! o   +
+       !   o +
+             +
+ + + + + + + +

s
+ + + + + + + +
+   .         +
+     i ! o   +
+       !   o +
+             +
+ + + + + + + +

d
+ + + + + + + +
+   .         +
+       i .   +
+       !   o +
+             +
+ + + + + + + +

a
+ + + + + + + +
+   .         +
+     i   .   +
+       !   o +
+             +
+ + + + + + + +

s
+ + + + + + + +
+   .         +
+         .   +
+     i !   o +
+             +
+ + + + + + + +

d
+ + + + + + + +
+   .         +
+         .   +
+       i ! o +
+             +
+ + + + + + + +

d  (Player satisfies last box, satisfying the win condition and
terminating the game)
+ + + + + + + +
+   .         +
+         .   +
+         i . +
+             +
+ + + + + + + +
You Win!
```

415774.2960412.qx3zqy7

LAB ACTIVITY | 12.1.1: ZyProject1: Sokoban | 0 / 18

Current file: **main.py** Load default template...

```python
from game_settings import *
for line in board:
    print(' '.join(line))
game_input = input()
while game_input in CONTROLS:
    if game_input == QUIT:
        print("Goodbye")
        break
    #if sokoban_input == 'w':
    #if sokoban_input == 's':
    if game_input == 'a':
        for line in board:
            for obj in range(len(line)):
                if line[obj] == SPRITE and line[obj-1] == EMPTY:
                    line[obj] = EMPTY
                    line[obj-1]= SPRITE
    if game_input == 'd':
        for line in board:
```

Develop mode **Submit mode**

Run your program as often as you'd like, before submitting for grading. Below, type any needed input values in the first box, then click **Run program** and observe the program's output in the second box.

Predefine program input (optional)

```
W
```

Run my program   Stop   Clear terminal

```
>
```

Coding trail of your work   What is this?

```
11/16 W----------------------------- min:21
```

Trouble with lab?