# **CS 1: Introduction to Computer Programming**

# Recitation 3: Nested for loops, while loops, type annotations Solutions

In today's syntax recitation, we'll cover use cases for while loops, practice using type annotations, and do some more practice with for loops!

# **Problem Solutions**

### Password Suggester (topics: Looping over strings, while loops)

Let's build a password suggester together. Start by implementing check\_password by following its docstring. Fill in the respective type annotations as well!

#### Valid Password (topic: looping over strings)

1	<b>def</b> check_	passw	ord(password:	str	) ->	1	bool	:		
2										
3	Checks	wheth	whether a password contains at least 3 vowels (A, E, I, O, U), case—insensitive.							
4										
5	Args:									
6	pass	password (???): The password string to validate.								
7 8	Returns:									
9										
10	???: True if the password has 3 or more vowels, False otherwise.									
11	count: int = 0									
		. [						]		
12	<b>for</b> cha	ar <b>in</b>		password.u	pper()			:		
	[	l					]	J		
13	if		cha	ar in "AEIOU"			:			
14	(	count ·	+= 1							
15	return			count >= 3						

#### Solution:

Alternatively, you can do char in "AEIOUaeiou", or replace "AEIOU" with ["A", "E", "I", "O", "U"] or set(["A", "E", "I", "O", "U"]).

#### Password Suggester (topic: while loops)

```
def prompt_for_passwords(num_passwords: int) -> None:
1
2
      Queries the user until we have num_passwords valid passwords.
3
4
      A password is valid if it contains at least 3 vowels.
5
6
      Args:
         num_passwords (int): The number of valid passwords to collect.
7
       .....
8
9
      valid_count: int = 0
      while
10
                                                                    :
                         valid_count < num_passwords</pre>
          pwd: str = input("Enter a password: ")
11
12
          if
                                                                    :
                             check_password(pwd)
             print("Valid password:", pwd)
13
```

14	<pre>valid_count += 1</pre>						
15	else:						
16	<pre>print("Invalid password. Try again.")</pre>						

### Loop-de-loop (topics: for loops, tuples, sets)

Fill in the relevant type annotations for each block of code. Note what each block of code outputs. If there is an error, write "Error".

#### Code

```
lst: list[str] = ["hi", "hello", "hey"]
for elem in lst:
   if len(elem) > 2:
      print(elem)
```

#### Solution:

hello

hey

Code

#### 1 2 3 4

lst: list[str] = ["hi", "hello", "hey"]
for elem in lst:
 elem = "LOL"
print(lst[0])

### Solution:

hi

```
1
2
3
4
```

1

2 3

```
Code
    lst: list[str] = ["hi", "hello", "hey"]
    for i in range(len(lst)):
        lst[i] = "LOL"
    print(lst[0])
```

#### Solution:

LOL

```
        tup:
        tuple[str, str, str]
        = ("hi", "hello", "hey")

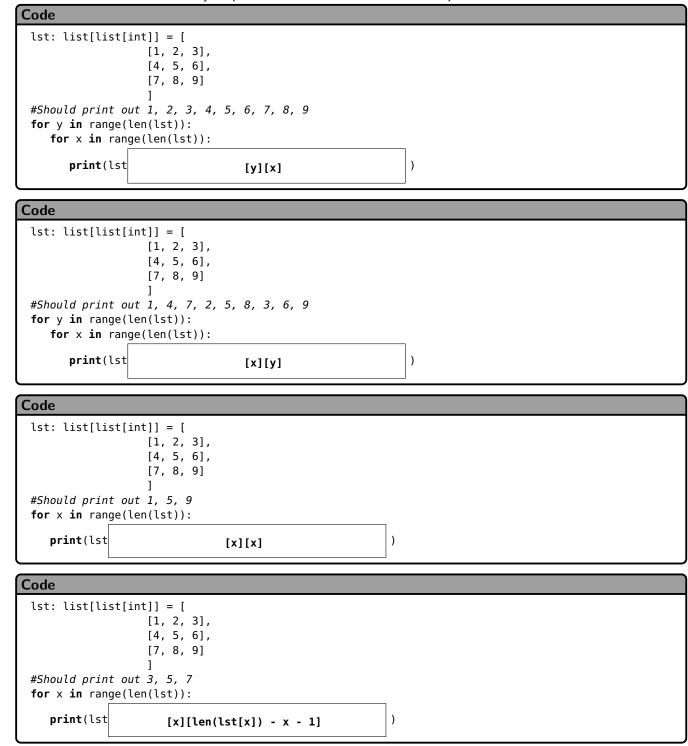
        tup[0] = "LOL"
        print(tup[0])
```

### Solution:

Error. Cannot change values in a tuple.

### It's like a maze...

Fill in the line of code necessary to print out all elements in the order specified.



#### Solution:

Since this is a square board, len(lst) would also work.