Python les-materialen

# Warm Up Project Exercises

It is time to get you to put together all your skills to start building usable projects! Before you jump into our full milestone project, we will go through some warm-up component exercises, to get you comfortable with a few key ideas we use in the milestone project and larger projects in general, specifically:

* Getting User Input
* Creating Functions that edit variables based on user input
* Generating output
* Joining User Inputs and Logic Flow

## Function to Display Information

**Creating a function that displays a list for the user**

def display\_list(mylist):  
 print(mylist)

mylist = [0,1,2,3,4,5,6,7,8,9,10]  
display\_list(mylist)

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

## Accepting User Input

\*\*Creating function that takes in an input from user and returns the result in the correct data type. Be careful when using the input() function, running that cell twice without providing an input value will cause python to get hung up waiting for the initial value on the first run. You will notice an In[\*] next to the cell if it gets stuck, in which case, simply restart the kernel and re-run any necessary cells.\*\*

input('Please enter a value: ')

Please enter a value: 2  
  
  
  
  
  
'2'

result = input("Please enter a number: ")

Please enter a number: 2

result

'2'

type(result)

str

int(result)

2

result = int(input("Please enter a number: "))

Please enter a number: 2

type(result)

int

# Example of an error!  
result = int(input("Please enter a number: "))

Please enter a number: two  
  
  
  
---------------------------------------------------------------------------  
  
ValueError Traceback (most recent call last)  
  
<ipython-input-19-202dd8101f66> in <module>()  
 1 # Example of an error!  
----> 2 result = int(input("Please enter a number: "))  
  
  
ValueError: invalid literal for int() with base 10: 'two'

\*\* Creating a function to hold this logic: \*\*

def user\_choice():  
 '''  
 User inputs a number (0-10) and we return this in integer form.  
 No parameter is passed when calling this function.  
 '''  
 choice = input("Please input a number (0-10)")  
   
 return int(choice)

user\_choice()

Please input a number (0-10)2  
  
  
  
  
  
2

result = user\_choice()

Please input a number (0-10)2

result

2

type(result)

int

## Validating User Input

\*\* Check that input is valid before attempting to convert.\*\*

We’ll use a simple method here.

As you get more advanced, you can start looking at other ways of doing this (these methods will make more sense later on in the course, so don’t worry about them for now).

* [Various Posts on This](https://www.google.com/search?q=python+check+if+input+is+number)
* [StackOverflow Post 1](https://stackoverflow.com/questions/5424716/how-to-check-if-string-input-is-a-number)
* [StackOverflow Post 2](https://stackoverflow.com/questions/1265665/how-can-i-check-if-a-string-represents-an-int-without-using-try-except)

some\_input = '10'

# Lot's of .is methods availble on string  
some\_input.isdigit()

True

\*\* Edit the function to confirm against an acceptable value or type \*\*

def user\_choice():  
   
 # This original choice value can be anything that isn't an integer  
 choice = 'wrong'  
   
 # While the choice is not a digit, keep asking for input.  
 while choice.isdigit() == False:  
   
 # we shouldn't convert here, otherwise we get an error on a wrong input  
 choice = input("Choose a number: ")  
   
 # We can convert once the while loop above has confirmed we have a digit.  
 return int(choice)

user\_choice()

Choose a number: hello  
Choose a number: two  
Choose a number: 2  
  
  
  
  
  
2

**Let’s try adding an error message within the while loop!**

def user\_choice():  
   
 # This original choice value can be anything that isn't an integer  
 choice = 'wrong'  
   
 # While the choice is not a digit, keep asking for input.  
 while choice.isdigit() == False:  
   
 # we shouldn't convert here, otherwise we get an error on a wrong input  
 choice = input("Choose a number: ")  
   
 # Error Message Check  
 if choice.isdigit() == False:  
 print("Sorry, but you did not enter an integer. Please try again.")  
   
 # We can convert once the while loop above has confirmed we have a digit.  
 return int(choice)

user\_choice()

Choose a number: two  
Sorry, but you did not enter an integer. Please try again.  
Choose a number: 2  
  
  
  
  
  
2

**Now let’s explore how to “clear” the output, that way we don’t see the history of the “Choose a number” statements.**

**NOTE: Jupyter Notebook users will use the IPython method shown here. Other IDE users (PyCharm, VS, etc..) will use**

from IPython.display import clear\_output  
clear\_output()

def user\_choice():  
   
 # This original choice value can be anything that isn't an integer  
 choice = 'wrong'  
   
 # While the choice is not a digit, keep asking for input.  
 while choice.isdigit() == False:  
   
 # we shouldn't convert here, otherwise we get an error on a wrong input  
 choice = input("Choose a number: ")  
   
 if choice.isdigit() == False:  
 # THIS CLEARS THE CURRENT OUTPUT BELOW THE CELL  
 clear\_output()  
   
 print("Sorry, but you did not enter an integer. Please try again.")  
   
   
 # Optionally you can clear everything after running the function  
 # clear\_output()  
   
 # We can convert once the while loop above has confirmed we have a digit.  
 return int(choice)

user\_choice()

Choose a number: 2  
  
  
  
  
  
2

**Checking Against Multiple Possible Values**

result = 'wrong value'  
acceptable\_values = ['0','1','2']

result in acceptable\_values

False

result not in acceptable\_values

True

from IPython.display import clear\_output  
clear\_output()

def user\_choice():  
   
 # This original choice value can be anything that isn't an integer  
 choice = 'wrong'  
   
 # While the choice is not a digit, keep asking for input.  
 while choice not in ['0','1','2']:  
   
 # we shouldn't convert here, otherwise we get an error on a wrong input  
 choice = input("Choose one of these numbers (0,1,2): ")  
   
 if choice not in ['0','1','2']:  
 # THIS CLEARS THE CURRENT OUTPUT BELOW THE CELL  
 clear\_output()  
   
 print("Sorry, but you did not choose a value in the correct range (0,1,2)")  
   
   
 # Optionally you can clear everything after running the function  
 # clear\_output()  
   
 # We can convert once the while loop above has confirmed we have a digit.  
 return int(choice)

user\_choice()

Choose one of these numbers (0,1,2): 1  
  
  
  
  
  
1

### More Flexible Example

def user\_choice():  
   
 choice ='WRONG'  
 within\_range = False  
   
 while choice.isdigit() == False or within\_range == False:  
   
   
   
 choice = input("Please enter a number (0-10): ")  
   
 if choice.isdigit() == False:  
 print("Sorry that is not a digit!")  
   
 if choice.isdigit() == True:  
 if int(choice) in range(0,10):  
 within\_range = True  
 else:  
 within\_range = False  
   
   
 return int(choice)

user\_choice()

Please enter a number (0-10): 12  
Please enter a number (0-10): 2  
  
  
  
  
  
2

## Simple User Interaction

**Finally let’s combine all of these ideas to create a small game where a user can choose a “position” in an existing list and replace it with a value of their choice.**

game\_list = [0,1,2]

def display\_game(game\_list):  
 print("Here is the current list")  
 print(game\_list)

display\_game(game\_list)

Here is the current list  
['hi', 'no', 2]

def position\_choice():  
   
 # This original choice value can be anything that isn't an integer  
 choice = 'wrong'  
   
 # While the choice is not a digit, keep asking for input.  
 while choice not in ['0','1','2']:  
   
 # we shouldn't convert here, otherwise we get an error on a wrong input  
 choice = input("Pick a position to replace (0,1,2): ")  
   
 if choice not in ['0','1','2']:  
 # THIS CLEARS THE CURRENT OUTPUT BELOW THE CELL  
 clear\_output()  
   
 print("Sorry, but you did not choose a valid position (0,1,2)")  
   
   
 # Optionally you can clear everything after running the function  
 # clear\_output()  
   
 # We can convert once the while loop above has confirmed we have a digit.  
 return int(choice)

def replacement\_choice(game\_list,position):  
   
 user\_placement = input("Type a string to place at the position")  
   
 game\_list[position] = user\_placement  
   
 return game\_list

def gameon\_choice():  
   
 # This original choice value can be anything that isn't a Y or N  
 choice = 'wrong'  
   
 # While the choice is not a digit, keep asking for input.  
 while choice not in ['Y','N']:  
   
 # we shouldn't convert here, otherwise we get an error on a wrong input  
 choice = input("Would you like to keep playing? Y or N ")  
  
   
 if choice not in ['Y','N']:  
 # THIS CLEARS THE CURRENT OUTPUT BELOW THE CELL  
 clear\_output()  
   
 print("Sorry, I didn't understand. Please make sure to choose Y or N.")  
   
   
 # Optionally you can clear everything after running the function  
 # clear\_output()  
   
 if choice == "Y":  
 # Game is still on  
 return True  
 else:  
 # Game is over  
 return False

**Game Logic All Together**

# Variable to keep game playing  
game\_on = True  
  
# First Game List  
game\_list = [0,1,2]  
  
  
  
while game\_on:  
   
 # Clear any historical output and show the game list  
 clear\_output()  
 display\_game(game\_list)  
   
 # Have player choose position  
 position = position\_choice()  
   
 # Rewrite that position and update game\_list  
 game\_list = replacement\_choice(game\_list,position)  
   
 # Clear Screen and show the updated game list  
 clear\_output()  
 display\_game(game\_list)  
   
 # Ask if you want to keep playing  
 game\_on = gameon\_choice()

Here is the current list  
['34', 1, 'new value']  
Would you like to keep playing? Y or N N

**Great work! You now have an understanding of bringing functions and loop logics together to build a simple game. This will be expanded upon in the Milestone project!**