

# Project 7

[Start Assignment](#)

---

<b>Due</b>	Sunday by 11:59pm	<b>Points</b>	50	<b>Submitting</b>	a file upload	<b>File Types</b>	zip
------------	-------------------	---------------	----	-------------------	---------------	-------------------	-----

---

## Goals

At the completion of this assignment, you should:

- have gained additional experience working with lists
- have gained additional experience working with input, strip() function and while loops
- have gained additional experience working with conditionals, logical and Boolean operators, and break
- have gained additional experience using functions
- have gained experience using string isdigit() function
- have gained additional experience creating more complex applications

## Expected Output and Grading Rubric

You can view the expected output and grading rubric for this lab at the end of the assignment.

## Getting Started

For this assignment you will create a simple list manager program. Your list manager program will support the following use cases:


- Add data to a list
- Delete individual list items
- Clear the entire list
- Output available commands

The assignment contains the required functions, and a general outline to follow to implement the list manager program. As usual you are encouraged to follow the incremental development steps, including incremental testing.

## Assignment

### Part I: Required Functions

Create a file called **p7-list-manager.py** for your list manager program.

Below is a list of **required** functions, where *t* is a list parameter. You should create all of these functions in your code file with a single statement *pass* (see the Python online documentation for [pass](#) 

([https://docs.python.org/3.2/reference/simple\\_stmts.html#the-pass-statement](https://docs.python.org/3.2/reference/simple_stmts.html#the-pass-statement)), course text, or course lecture notes, if you forgot the purpose of `pass`).

1. `cmd_help()`
2. `cmd_add(t)`
3. `cmd_delete(t)`
4. `cmd_list(t)`
5. `cmd_clear(t)`
6. `get_max_list_item_size(t)`

You will also need to use your padding functions from a previous assignment. Copy all three functions into your code file.

- `pad_string()`
- `pad_left()`
- `pad_right()`

## Part II: List variable

You will require a list variable to hold your list items. The list item should be declared outside of any function, and initially be empty.

**Important:** For this assignment, your list item variable is essentially a "global" variable, but we've yet to deal with how to formally work with global variables. For this assignment, all of your functions that need to manipulate the list must accept the list as a parameter. You are allowed to directly modify the list parameter within functions, which means that the contents of the global list will also be changed. Below is an example of using the list variables.

```
def cmd_add(t):  
    ....  
  
my_list = []  
while .....  
    cmd_add(my_list)  
    .....
```

## Part III: Primary Input Loop

You will require a primary input loop to prompt for list commands. The loop must only handle the commands, and then call the appropriate function to handle the command. You must use the `strip()` function to remove leading and trailing white space. If the Enter key is pressed with no input, exit the program.

The prompt must be: **"Enter a command (? for help): "**.

## Part IV: Help Command

Your list manager must display help.

1. If the ? is entered as a command, display a list of commands, and a command description.
2. The commands must be contained within a list.
3. The command descriptions must be contained within a separate list.
4. These two lists must contain the same number of items.
5. Use the **get\_max\_list\_item\_size()** function to find the maximum length of all of the commands.
6. Use a for loop to iterate through the loops to display the help listing.
7. Use the **pad\_right()** function and the maximum command length to format the command, with a space between the command and the command description of five (5) spaces.
8. Add "Empty to exit" as a final help tip.

See the sample output below.

```
Enter a command (? for help): ?
*** Available commands ***
Add           Add to list
Delete        Delete information
List          List information
Clear         Clear list
Empty to exit
Enter a command (? for help): list
List contains 0 item(s)
Enter a command (? for help):
Goodbye!
```

## Part V: Add Command

Your list manager program must support adding data to the list.

1. The add command will prompt the user for data to add until the Enter key is pressed with no input.
2. After each data is entered, display a notification that data was added, and a count of the total number of list items.

See the sample output below.

```
Enter a command (? for help): add
Enter information (empty to stop): Item1
Added, item count = 1
Enter information (empty to stop): Item2
Added, item count = 2
Enter information (empty to stop): Item3
Added, item count = 3
Enter information (empty to stop): Item4
Added, item count = 4
Enter information (empty to stop):
Enter a command (? for help): list
List contains 4 item(s)
Item1
Item2
Item3
Item4
```

## Part VI: List Command

The list command will display the total number of items in the list, and each item in the list.

See the image above in Part V and the add command for sample output.

## Part VII: Clear Command

The clear command will clear (delete) all items in the list, and specify how many items were removed and that the list is now empty.

You may use the list *clear()* function to empty the list.

See the sample output below.

```
Enter a command (? for help): list
List contains 0 item(s)
Enter a command (? for help): clear
0 item(s) removed, list empty
Enter a command (? for help): add
Enter information (empty to stop): Item1
Added, item count = 1
Enter information (empty to stop):
Enter a command (? for help): clear
1 item(s) removed, list empty
Enter a command (? for help):
```

## Part VIII: Delete Command

The delete command will remove items from the list. Your list manager must support the shortcut del command as well.

1. The delete command will display all items with the index number next to each item.
2. Use the `pad_right()` function to format the index column, with a space of two (2) blank spaces between the index and the list item value.
3. The delete command will continuously prompt for data to delete until either all data is deleted, or the Enter key is pressed with no input.
4. The delete command must validate the entered index is a valid integer, and the value is in the range appropriate for the number of items in the list.
5. Use the `strip()` and string [`isdigit\(\)`](https://www.w3schools.com/python/ref_string_isdigit.asp) function to test if the entered index is a valid integer.
6. Output appropriate error messages if incorrect data is entered.

See the sample output below.

```
Enter a command (? for help): list
List contains 4 item(s)
Item1
Item2
Item3
Item4
Enter a command (? for help): del
0  Item1
1  Item2
2  Item3
3  Item4
Enter number to delete (empty to stop): 3
0  Item1
1  Item2
2  Item3
Enter number to delete (empty to stop): 2
0  Item1
1  Item2
Enter number to delete (empty to stop): 1
0  Item1
Enter number to delete (empty to stop): 0
All items deleted
Enter a command (? for help): list
List contains 0 item(s)
Enter a command (? for help):
```

---

## Submissions

For this assignment you will be uploading your work from Lab 7 as well as your work from this assignment.

*Remember that all submitted **Python** files must include the header docstring at the top of each file.*

Create a single compressed file that contains the Python files. To create the compressed file, select the deliverable Python files, right-mouse click on the selected files, and select Compress (Mac) or Compressed (Windows). A new file will be created that contains all of the deliverable Python files.

Rename the compressed file **p7.zip**. Use the Project 7 Canvas site to submit/upload the compressed/ZIP file.

The compressed file will contain the following deliverables (point values in parentheses).

- **(5) lab8-lists.py**
- **(45) p7-list-manager.py**

Once uploaded, revisit the Project 7 page in Canvas and confirm the file was uploaded.

**Remember, you are responsible to submit your assignments before the deadline. Late submissions will not be accepted.**

## Expected Output

The expected output for this project is described in the project definition.

## Grading Rubric

### p7-list-manager.py

- Verify cmd\_help() function implemented and called
- Verify cmd\_add() function implemented and called
- Verify cmd\_delete() function implemented and called
- Verify cmd\_list() function implemented and called
- Verify cmd\_clear() function implemented and called
- Verify get\_max\_list\_item\_size() function implemented and called
- Verify pad\_left() function implemented/imported
- Verify pad\_right() function implemented/imported and called
- Verify using a command prompt
- Verify using strip()
- Verify using isdigit()
- Verify validating input
- Validate ? returns help
- Validate add command functionality
- Validate delete command functionality

- Validate list command functionality
- Validate clear command functionality