11.5 "Update" menu option

Add the following branch to your **main program**; replace the ellipses with the appropriate values:

```
    elif opt == 'U':
        continue_action = 'y'
        while continue_action == 'y':
            if ... == []: # TODO
                print("WARNING: There is nothing to update!")
                break
            print("::: Which task would you like to update?")
            print_tasks(all_tasks, priority_scale, name_only = True, show_idx = True, start_idx = 1)
            print("::: Enter the number corresponding to the task.")
            user_option = input("> ")
            if ...: # TODO
                ... # TODO: convert the index appropriately to account for the start_idx = 1
                subopt = get_selection("update", all_tasks[...], to_upper = False, go_back = True)
                if subopt == 'M': # if the user changed their mind
                    break
                print(f"::: Enter a new value for the field |{...}|") # TODO
                field_info = input("> ")
                result = update_task(all_tasks, user_option, priority_scale, subopt, field_info, start_idx = 1)
                if type(result) == dict:
                    print(f"Successfully updated the field |{...}|:")  # TODO
                    print_task(result, ...)  # TODO
                else: # update_task() returned an error
                    print(f"WARNING: invalid information for the field |{...}|!")  # TODO
                    print(f"The task was not updated.")
            else: # is_valid_index() returned False
```

```
                print(f"WARNING: |{...}| is an invalid task
number!")  # TODO

            print("::: Would you like to update another task?",
end=" ")
            continue_action = input("Enter 'y' to continue.\n>
")
            continue_action = continue_action.lower()
            # --------------------------------------------------
---------------
```

You will need the `is_valid_index()` function that was introduced in LAB 8.12. Below is its updated/clarified documentation:

```
def is_valid_index(idx, in_list, start_idx = 0):
    """
    param: idx (str) - a string that is expected to
            contain an integer index to validate
    param: in_list - a list that the idx indexes
    param: start_idx (int) - by default, set to 0;
            an expected starting value for idx that
            gets subtracted from idx for 0-based indexing

    The function checks if the input string contains
    only digits and verifies that (idx - start_idx) is >= 0,
    which allows to retrieve an element from in_list.

    returns:
    - True, if idx is a numeric index >= start_idx
    that can retrieve an element from in_list.
    - False if idx is not a string that represents an
    integer value, if int(idx) is < start_idx,
    or if it exceeds the size of in_list.
    """
```

Define a new function `update_task()` to update the task list appropriately:

```
def update_task(info_list, idx, priority_map, field_key,
field_info, start_idx = 0):
    """
    param: info_list - a list that contains task dictionaries
    param: idx (str) - a string that is expected to contain an
integer
            index of an item in the input list
    param: start_idx (int) - by default is set to 0;
            an expected starting value for idx that gets
subtracted
```

```
            from idx for 0-based indexing
    param: priority_map (dict) - a dictionary that contains the
mapping
            between the integer priority value (key) to its
representation
            (e.g., key 1 might map to the priority value
"Highest" or "Low")
            Needed if "field_key" is "priority" to validate its
value.
    param: field_key (string) - a text expected to contain the
name
            of a key in the info_list[idx] dictionary whose
value needs to
            be updated with the value from field_info
    param: field_info (string) - a text expected to contain the
value
            to validate and with which to update the dictionary
field
            info_list[idx][field_key]. The string gets stripped
of the
            whitespace and gets converted to the correct type,
depending
            on the expected type of the field_key.

    The function first calls one of its helper functions
    to validate the idx and the provided field.
    If validation succeeds, the function proceeds with the
update.

    return:
    If info_list is empty, return 0.
    If the idx is invalid, return -1.
    If the field_key is invalid, return -2.
    If validation passes, return the dictionary info_list[idx].
    Otherwise, return the field_key.

    Helper functions:
    The function calls the following helper functions:
    - is_valid_index()
    Depending on the field_key, it also calls:
    - is_valid_name()
    - is_valid_priority()
    - is_valid_date()
    - is_valid_completion()
    """
```

LAB ACTIVITY | 11.5.1: "Update" menu option 0 / 1

main.py

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.

Enter program input (optional)

If your code requires input values, provide them here.

Run program

Input (from above) → **main.py** (Your program)

Program output displayed here

Coding trail of your work What is this?

```
History of your effort will appear here once you begin working on this zyLab.
```

Trouble with lab?