Lab 04 Specification – A Hand-on Exercise to practice Dynamic Arrays
50 points

Due by: 04/21/2021 8:00 AM

Lab Goals

- Learn to develop Dynamic Array structure.
- Do a simple exercise and conduct experiments to compare Dynamic Arrays, Built-in Arrays, and List to get insights on their efficiency.

Learning Assignment

If you have not done so already, please read all of the relevant ”GitHub Guides”, available at the following website:
`https://guides.github.com/`
that explains how to use many of the features that GitHub provides. This reading will help you to understand how to use both GitHub and GitHub Classroom. To do well on this assignment, you should also read:

- **GT chapter 05, 5.2,5.3,5.4**

Assignment Details

Now that we have discussed some basics of Dynamic data structures (Dynamic Arrays) together in the last few lectures, it is your turn. In this lab, you will practice a variety of programs to retain the knowledge of Dynamic data structures that had been covered so far. This includes modifying one or more code files to implement a series of functionalities. At any duration during and/or after the lab, students are recommended to team up with the Professor and the TL(s) to clarify if there is any confusion related to the lab and/or class materials.

Students are recommended to get started with this part in the laboratory session, by discussing ideas and clarifying with the Professor and the Technical Leader(s). It is acceptable to discuss high-level ideas with your peers, while all the work should be done individually. Late submission is accepted for the part(s) in this section, based on the late policy outlined in the course syllabus.

It is required for all students to follow the honor code. Some important points from the class honor code are outlined below for your reference:

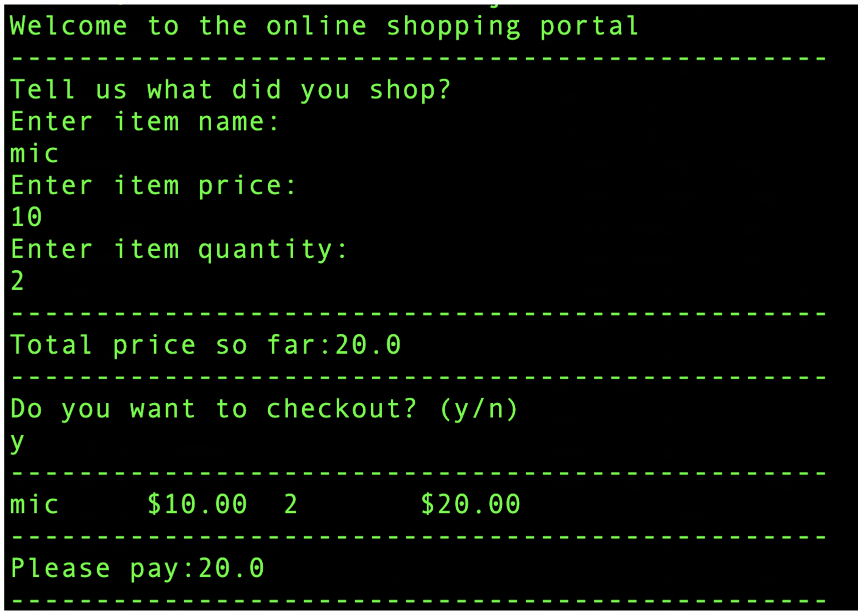
1. Students are not allowed to share code files and/or other implementation details. It is acceptable to have a healthy discussion with your peers. However, this discussion should be limited to sharing ideas only.
2. Submitting a copy of the other’s program(s) is strictly not allowed. Please note that all work done during lab sessions will be an opportunity for students to learn, practice, and master the materials taught in this course. By doing the work individually, students maximize the learning and increase the chances to do well in other assessments such as skill test, exams, etc $\cdots$

Part 01 - Developing a Shopping Cart Application (50 points)

In this part, you will develop a Shopping Cart application. The application mainly uses a Dynamic array in conjunction with an object-oriented technique to store and process the items shopped by customers. To simplify the development process, a series of starter-code is provided in the repository, with the file named `item.py`, `shoppingcart.py`, and `checkout.py`. Please make sure to follow the requirements outlined below:

1. Complete the Item class file. A constructor to initialize all the members of the class is provided in the starter code. The __str__() method is provided in the stater-code to stringify the results to be displayed to the user based on the item properties. Add the respective getters for all the three private members in the class.
2. Complete the shoppingcart class by doing the following:
   - A constructor to the class is provided in the starter code. The capacity should be set to 5. Please refer to darray.py. Similar to this code file, we initialize the `cart` array of size equal to the capacity. The idea behind this initialization is that the array of items is set to be of size 5 and will have its size increased once the capacity is reached.
   - The method `getTotalPrice()` is already implemented in the starter code. This method simply is a getter for the `totalPrice` member variable.
   - The method `displayItemsDuringCheckout` displays all the items in the cart and their respective information such as price, quantity, and name. This method makes a call to the __str__() method in the Item class.
   - The method `addItemToCart` is incomplete in the starter-code. Please add the required logic to add an item to the cart. In this process, it is first required to create an object for each individual item shopped. Recall that a variable to hold array of objects of type Item is already created in the earlier step named `cart`. After display the item details on the console, the variable `itemCount` and `totalPrice` should be incremented. This increment is done so to compute the `totalPrice` and the `itemCount` for every item shopped. If the `itemCount` reaches the capacity, then a call should be to the increaseCartSize() method.
   - The method `increaseCartSize` method is incomplete in the starter-code. Please add the required logic in this method to take the `cart` array and make it bigger by increasing the size by 5 every time. The capacity was originally 5 and every time this method is called the `cart` size should increase by 5. How to do this? Refer to the darray.py file. Create a new array called `temp` with its size as (capacity + 5). Move all the elements from the `cart` array to the `temp` array. Then reinitialize the `cart` array to the increased size. Note: at this point the `cart` array will be empty (because it is reinitialized) and now assign `cart` = `temp`. Finally, make sure to increment the capacity by adding 5 to it. In this way, the new capacity is 5 more than the old capacity value. This is a critical part of the lab. It is important to understand how to change the size of an array. An array is fixed in size. Hence, we have a huge limitation to add new elements to the array. Completing this part will ensure that a student will fully understand the concept of Dynamic Arrays and an array of Objects.
3. The checkout class is complete in the starter-code. The details related to the `checkout` class is purposefully left out from this sheet. It is required to analyze the checkout class in order to fully implement the other parts of this application. There are no code changes required to be done in the checkout class. If all the incomplete code is implemented correctly, then the program should print the results.

4. A sample output is displayed below for your reference:



```
Welcome to the online shopping portal
-------------------------------------------------
Tell us what did you shop?
Enter item name:
mic
Enter item price:
10
Enter item quantity:
2
-------------------------------------------------
Total price so far:20.0
-------------------------------------------------
Do you want to checkout? (y/n)
y
-------------------------------------------------
mic      $10.00  2       $20.00
-------------------------------------------------
Please pay:20.0
-------------------------------------------------
```

You are not allowed to use a list, array, numpy array to implement this part. You can only use Dynamic Arrays.

Reason for that is to practice Dynamic Arrays.

Part 02 - Honor Code

Make sure to **Sign** the following statement in the `honor-code.txt` file in your repository. To sign your name, simply replace Student Name with your name. The lab work will not be graded unless the honor code file is signed by you.

This work is mine unless otherwise cited - Student Name

Submission Details

For this assignment, please submit the following to your GitHub repository by using the link shared to you by the Professor:

1. `item.py`, `checkout.py`, and `shoppingcart.py` file.
2. A signed honor code file, named `honor-code.txt`.
3. To reiterate, it is highly important, for you to meet the honor code standards provided by the college. The honor code policy can be accessed through the course syllabus.

Grading Rubric

1. There will be full points awarded for the lab if all the requirements in the lab specification are correctly implemented. Partial credits will be awarded if deemed appropriate.
2. Failure to upload the lab assignment code to your git repo will lead you to receive no points given for the lab submission. In this case, there is no solid base to grade the work.
3. There will be no partial credit awarded if your code doesn't compile correctly. It is highly recommended to validate if the correct version of the code is being submitted before the due date and make sure to follow the honor code policy described in the syllabus. If it is a late submission, then it is the student's responsibility to let the professor know about it after the final submission in GitHub. In this way, an updated version of the student's submission will be used for grading. If the student did not communicate about the late submission, then automatically, the most updated version before the submission deadline will be used for grading purposes. If the student had not submitted any code, then, in this case, there are no points awarded to the student.
4. If you need any clarification on your lab grade, talk to the Professor. The lab grade may be changed if deemed appropriate.