**Todo Application**

**Documentation for Application**

Simple TodoApplication for keeping track of daily routines that will be able to categorize them into 4 different classes. We will see this part later in the documentation.

Things that we will need to have to start building the application: Basic git command line usage and GitHub to upload and commit our code at the end.

How to install and execute the application for it to work on an operating system.

For this application we will be using two libraries that we will be importing from python, one of them will be pickle library and the other will be system library which is used at the start of the application.

To begin with, we will need to install python IDE on the operating system on which we will start building our application.

We can also use depending on what is preferred and easier. Start by opening terminal and run it on terminal and save it as python Todo.py or run it on an IDE which we will be performing the task which are given on the assignment.

The assignment project application will need to be able to do these four features which are given at the beginning of the project.

First will be a list of tasks on which we will be able to add or remove them from a list at any given time and be able to check for task that have been completed.

We will first open the file using python and the file name that we have created which will be “task.txt” that we will use to store the commands that we will give the application to add or remove.

For this python will load the library using “loadtsk” that will open pickle.load and see the given task that were added to the library.

The application will then check for arguments, if there are no arguments that are run or typed the application will directly show and display a menu, we will need to choose from one of the following:

Command-line arguments:  
 -l Lists all the tasks  
 -a Adds a new task  
 -r Removes a task  
 -c Completes a task

The calling function that will be added

We will check if the task description is provided after the -a parameter before adding it to the list and saving to file that will be used later.

elif "-l" in sys.argv

If the LoadedTasks variable is empty, we will provide the correct error message that will be display.

If it's empty it should display on the screen for you “No todos for today! :)” if not it should display “the task has not been completed yet” and should not be marked with an X as the one before.

elif "-r" in sys.argv

We will then need to be able to remove tasks from the application , for this we will use LoadedTasks.pop(int(TaskNumber) - 1) which will let us remove the task from the LoadedTasks list and  
pickle.dump(LoadedTasks, open('tasks.txt', 'wb')) which we will write again and verify that the file has been remove and save it.

If there is an error within the index it will have to know if its out of bound the value or if the index is not a number depending on the value that is entered.

elif "-c" in sys.argv

We will now see if the tasks has been completed or not for this we will use TaskNumber = sys.argv[2] to check the number 2 taks and also check if there is any errors on the written input.  
If there is an error, it should print "Unable to remove: no index provided".

If it is true it should open the task.txt file and see if any of the files have been completed or not and if there is any exception with the index if its out of bound or number.

If none of the arguments are listed on the task and none of the conditions match, then the wrong argument must be provided and should print "Unsupported argument".