Name of Use Case: Specify/Create Todo.txt file
Created by: Justin Davis
Date Created: 6/8/2021
Description: Allows a user to specify the name they want for the Todo.txt file, and if there isn't one that exist it creates it.
Actors: User
Preconditions: Since this works whether a file exists or not, there aren't any.
Postconditions: The file is chosen or created.

Flow:
   1. User enters the file name for the todo list.
   2. System checks if there is a file.
   3. If the file exists, it pulls the data from that list and allows the user to edit it.
   4. If the file doesn't exist, the system creates one and then opens it for edits.

Alternative Flows: 1.
Exceptions: 1.

Name of Use Case: Add Tasks
Created by: Justin Davis
Date Created: 6/8/2021
Description: Allows a user to specify a new task, and adds it to the current todo list.
Actors: User
Preconditions: File has been opened/created.
Postconditions: The desired task is added to the current todo.

Flow:
   1. User specifies the various parts of the todo they want to add.
   2. System adds the new task to the todo list.

Alternative Flows: 1.
Exceptions: 1.

Name of Use Case: View Todo List
Created by: Justin Davis
Date Created: 6/8/2021
Description: Allows a user to see the current todo list in its entirety.
Actors: User
Preconditions: File has been opened/created and there are tasks in the list.
Postconditions: The todo is printed to the console.

Flow:
   1. User requests to view the todo
   2. System prints out the todo

Alternative Flows: 1. List has no task in it (Exception 1)

Exceptions:
1.   System informs the user that the list is empty.

Name of Use Case: View Completed List
Created by: Justin Davis
Date Created: 6/8/2021
Description: Allows a user to view the list of completed tasks
Actors: User
Preconditions: File has been opened/created, and the list has tasks in it.
Postconditions: The completed list is printed to the screen.

Flow:
    1.   User requests to view the completed list.
    2.   System prints out the completed list to the console.

Alternative Flows: 1.   List has no task in it (Exception 1)
Exceptions: 1.   System informs the user they haven't completed anything and therefore the list is empty.

Name of Use Case: Update List Item
Created by: Justin Davis
Date Created: 6/8/2021
Description: Allows a user to update a task in the todo list.
Actors: User
Preconditions: File has been opened/created, the user states the task they want to update, and the list has tasks in it.
Postconditions: The system updates the desired task with the desired changes.

Flow:
    1.   User specifies the task they want to update.
    2.   System prompts them for the updated info and edits the current task with the new info.

Alternative Flows:
    1.   User specifies a task that doesn't exist (Exception 1)
    2.   List has no tasks in it (Exception 2)

Exceptions:
    1.   System informs the user that the task doesn't exist in the list.
    2.   System informs the user that the list is empty.

Name of Use Case: Check for existing todo.txt
Created by: Justin Davis
Date Created: 6/8/2021
Description: Allows the system to determine if the desired text file exists.
Actors: System
Preconditions: Since this works whether a file exists or not, there aren't any.
Postconditions: A file is either confirmed to exist or not.
Flow: 1.   System looks for a todo.txt with the given name.

2. Returns what it found.
Alternative Flows:
Exceptions:

Name of Use Case: Display Todo List
Created by: Justin Davis
Date Created: 6/8/2021
Description: System prints out the current todo list.
Actors: System
Preconditions: User has requested to view a todo list and no exceptions were thrown.
Postconditions: Chosen todo list is displayed in the console.
Flow: 　　1. System displays the currently opened todo list.
Alternative Flows:
Exceptions:

Name of Use Case: Display Completed List
Created by: Justin Davis
Date Created: 6/8/2021
Description: System prints out the completed list.
Actors: System
Preconditions: User has requested to view the completed list and no exceptions were thrown.
Postconditions: Completed list is printed to the screen.
Flow: 　　1. System displays the completed list.
Alternative Flows:
Exceptions:

Name of Use Case: Move Completed Task
Created by: Justin Davis
Date Created: 6/8/2021
Description: System moves a task that has been completed to the completed list.
Actors: System
Preconditions: User has updated a task with the completed marker.
Postconditions: Completed task has been removed from the todo, and into the completed list.
Flow: 　　1. User gave the update action a completed marker for a task. 　　2. System takes the task out of the todo list and inserts it into the completed list.
Alternative Flows:
Exceptions:

Todo

Specify/Create todo.txt file

Check for existing todo.txt

<<includes>>

Add Tasks

View Todo list

Display Todo list

<<includes>>

View completed list

Display Completed List

<<includes>>

Update list item

Move completed task

<<includes>>

User

System

List

+tasks: List<Tasks>

+InitializeTodo(fileName: string): void
+ToString():string
+AddTask(taskToAdd Task):void

Is-a

Is-a

TodoList

+RemoveTask(taskToRemove: Task):void
+UpdateTask(taskIndex: int,
    CompletedList completed):void

Task

+CompletionStatus:char
+Priority:char
+CompletionDate:string
+CreationDate:string
+ProjectTag:string[]
+ContextTag:string[]
+KeyValueTag:string

+Task(task: Task): ctor
+Task(completionStatus: string, priotiry:
    string, completionDate: string,
    creationDate: string, projectTag:
    List<string>, contextTag: List<string>,
    dueDate: Dictionary<string,string>)
+Equals(taskToCheck: Task): bool

CompletedList

ToString(): string

Has-a

1

1..*

Has-a

1

1..*