CSL100 - Assignment 5

June 30, 2014

Due Date: Monday July 7. 12:30 pm (afternoon)
Total Points: 20 (+ 4 Extra Credits)

In this assignment, we will design a personal task manager for the month of July 2014. For each day in the month, we would like to store a list of tasks to be done (referred to as *to do* tasks). We would like to be able to display the tasks for a given date, add new tasks, remove/delete tasks and save the changes to a file.

The maximum number of *to do* tasks for any day is 5. Initially, the list of tasks are read from a file. Each row in the file contains the date followed by a list of semi-colon separated tasks for that date. The following shows an example file:

```
1;Implement Assignment 5;Book Train Ticket
2;Call Home
3;Wash the Clothes
7;Submit the Assignment Online
8;Give Assignment Demo;Prepare for Major
10;Take the Exam
11;Go Home
```

Each task has an implicit id (between 1 and 5) associated with it based on the order in which it appears in the sequence of tasks for the day. As an example, for July 1st, 'Implement Assignment' has a task id of 1 associated with it and 'Book Train Ticket' has an id of 2. If a date does not have a row in the file associated with it (e.g. July 9), this means that task list for that date is empty. You will be implementing the task manager using a GUI (Graphical User Interface) in Python. More details on how to implement a GUI are provided at the end of this assignment. Figure 1 shows an example snapshot of the desired output. It supports the following operations.

1. Display the calendar for the month of July 2014.

2. Display the tasks for today's date in the area below the calendar.

3. When the user clicks on a date, the date is highlighted and the corresponding tasks are displayed (along with their associated ids).

4. To the right of calendar, there is a textbox to enter a new task to the list of *to do* tasks for the date currently being displayed. After entering the new task in the textbox, user needs to click on the 'ok' button to add it to the list. As soon as the 'ok' button is pressed, the list of

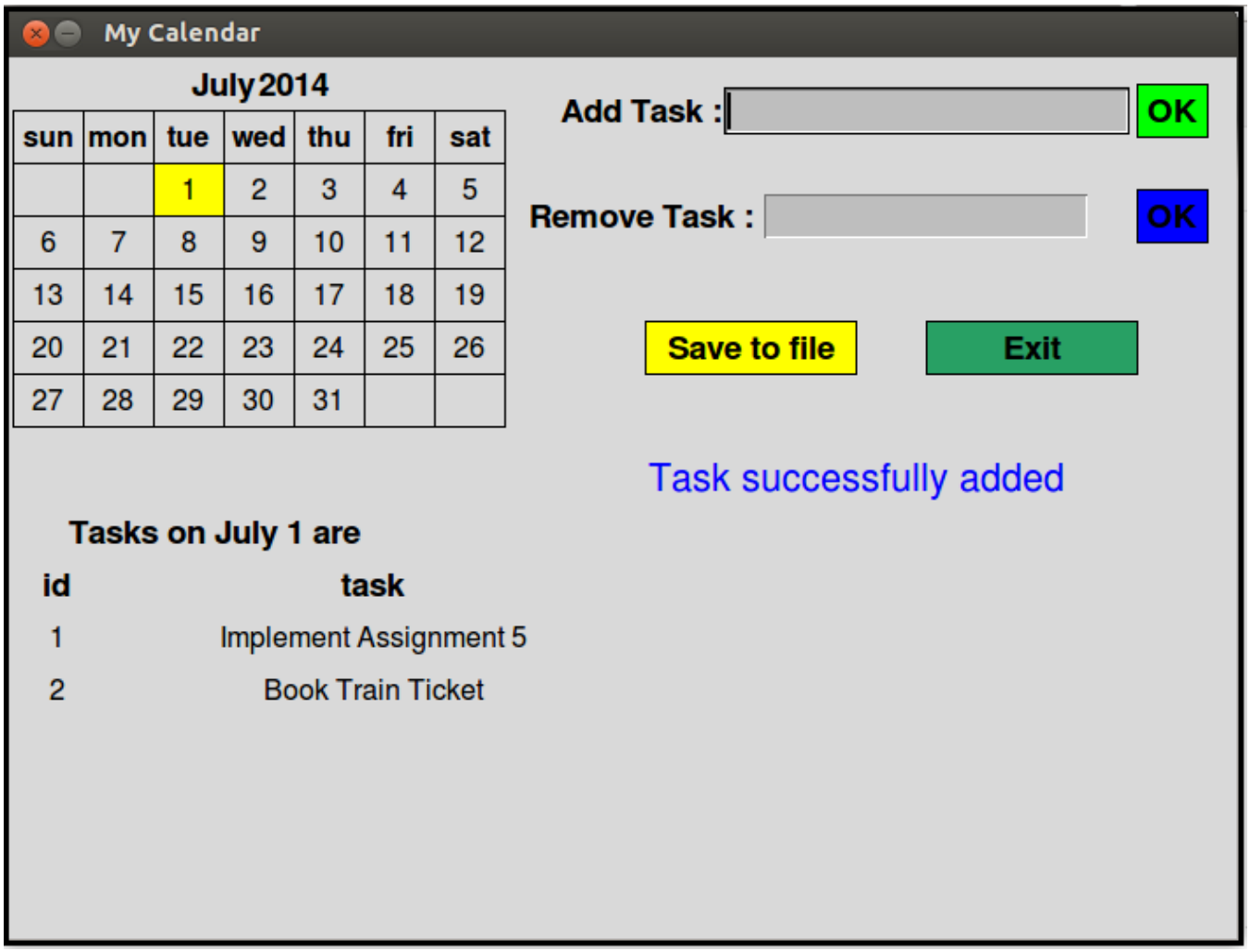

Figure 1: Example Task Manager

tasks being displayed should be refreshed to reflect the addition of the new task. The textbox should be cleared after the changes have been incorporated into the task list.

5. Similarly, there is a textbox to remove the task from the *to do* list for the date being displayed. The textbox takes as input the id of the task to be removed. As in case of addition, the user needs to click on the 'ok' button to remove a task.

6. There is a button 'Save to File' which saves the current changes to a file.

7. Clicking on the 'Exit' button exits the program.

In addition to supporting the above operations, you program should ensure the following:

1. Initial list of tasks should be read from a file named 'july-2014.txt'. Any saved changes should be written to the same file. If the file does not exist in the beginning, this means that the list of task is empty for each day. You should create a new file with this name and write any changes to this file.

2. The program should handle all kinds of incorrect inputs appropriately. For instance, if the user tries to add a task beyond 5 tasks, it should display an error message. Similarly, if the id of the task to be removed does not exist in the *to do* list, it should display an appropriate error message.

3. Before exiting, the program should confirm from the user if the latest changes need to be saved to the file (if not saved already). The changes should be saved to a file only if the user explicitly asks the system to do so.

4. In no case, the program should abruptly end and terminate.

5. Feel free to implement any creative displays of the task manager!

Extra Credit: Extend the task manager to work for all the months in the year 2014. Your program should allow for moving between months (for example, by having a forward/backward button at the top of the calendar), and allow for performing all the operations specified above for each of the months. For each month, you can store the tasks in a separate file. Before switching between months, the program should confirm from the user if the changes made to the current month should be saved to the corresponding file.

Implementing Graphical User Interface: Download this graphics package and store it in the directory where you will write your program. For a detailed documentation, click here. Here is a template file demonstrating the use of various utilities that you will need to write your program.