Name: _____

Please refer to the previous assignment for important instructions on the allowable use of resources and the requirements for electronic and hardcopy submission. Use the newsgroup for questions, clarifications, and general discussion related to this homework and class. Ensure that everything you submit is neat and easy to comprehend; otherwise, it will be ignored, with zero credit. You are encouraged to answer the three ⋆ questions for extra credit but do note that those questions are graded much more strictly than the others.

This assignment focuses on building a graphical user interface (GUI) using the *Swing* library, described in Appendix B of the textbook.[1] After studying some of the basics of Swing programming, the main task is to add a GUI to the simple record-manager implemented for Homework 3.

1. (1 pt.) Write your name in the space provided above.

2. (1 pt.) Package your solutions to the programming questions as in previous assignments. Submit your work via `http://cs.umaine.edu/~chaw/u/`. *After* submitting your work, complete the following:
   File name: _____ Size, in bytes: _____
   MD5 checksum: _____
   Timestamp: _____

3. (1 pt.) Identify and fix any errors in Figure B.2.

4. (1 pt.) Briefly explain the difference between lightweight and heavyweight components and the significance of the content pane.

5. (1 pt.) Briefly explain the difference between the `Component` methods `setSize` and `setPreferredSize`.

[1] Mark Allen Weiss, *Data Structures and Problem Solving Using Java*, 4th edition (Addison-Wesley, 2010).

6. (1 pt.) What are *listener adapter* classes? How are they used?

7. (4 pts.) An application requires users to either select one of the predefined names *Alice*, *Bob*, and *Cindy* or input a new name. Which Swing component is the most suitable? Provide a code fragment illustrating the creation of that component, with proper initialization.

8. (2 pts.) In an interface that includes several buttons, how may an event-handling method determine which button triggered the event? Provide a code fragment to illustrate your answer.

9. (3 pts.) An application that displays a computationally intensive animation would benefit from disabling the animation when the main application window is iconified. How may this feature be implemented? Provide a code fragment to illustrate your answer.

10. (5 pts.) In Figure B.5, the North and South buttons stretch all the way to the sides while the East and West buttons do not stretch all the way up and down. Provide code to reverse this situation, so that the East and West buttons occupy the entire right and left columns, while the North and South buttons are only as wide as the Center button.

11. (80 pts.) Recall the simple record-manager from Question 10 of Homework 2, based on an implementation of the *k-search-trees* of that homework. Using that implementation as the back-end, add a graphical front-end to the record-manager application as suggested below. You may modify your earlier k-search-tree implementation to fix bugs or add missing features, but you must use k-search-trees for all record-manager functionality. *Use the newsgroup for eliciting further details.*

    The basic interface permits choice of the desired *command* (cf. the table in Homework 2) using a pulldown menu. Arguments to the selected command, if any, are provided in suitably placed text input boxes. Commands are executed by activating a *Go* button. The text output of each command, as defined in Homework 2, appears in a suitably scrolling text area. In addition, each command's output should be immediately preceded by the text representation of the command in a different font and color. You are encouraged to augment this basic interface to provide a nicer user experience, but you should ensure that the basic functionality is supported.

12. $\star$ (20 pts.) Augment the interface to include a suitable graphical representation of the tree that is visible at all times.

13. $\star$ (15 pts.) Implement a Swing application that presents a window containing an analog clock, a digital clock, and a slider-based clock. All three components are meant to display the time of the day and should agree with one another, possibly except when the slider is being manipulated by the user. The slider represents time in 24-hour format and so ranges from 00:00:00 to 23:59:59, with markers every hour and half hour. When the application is launched, all three clocks should display the current time of the day, accurate to within a tenth of a second. When the slider-based clock is modified by the user, the other clocks should be updated to the newly set time in the slider, and all three clocks should continue to tick at the normal rate from that set time, which in general will differ from the time of the day. In addition, the application should include a button labeled *current time* that sets all three clocks to the current system time. The face of the analog clock should include a text indicator that reads a.m. or p.m. as appropriate. When the main application window is resized, all components should be resized to make the best use of the available space, with the analog clock given priority for space. That is, a giant clock is preferred to a giant slider, although the slider should also be expanded to make it easier to manipulate when space is available.

14. $\star$ (15 pts.) Modify the application of Question 13 to permit the time to be modified using not only the slider but also the analog clock, by dragging the hour, minute, and second hands with a mouse.