Is This the Happiest Place in America?

For the second time in a row, Gallup-Healthways ranked this Florida city first for overall well-being

BRIGIT KATZ
03/07/2017

Switzerland may be the best country in the world (or so says U.S. News & World Report), but there is plenty of happiness to be found here in the USA—
Announcements

- Homework 5 due Monday
- Midterm
  - NOT IN BRONFMAN
  - Evening of March 16th
  - At 6 p.m. or 7:30 p.m. in TPL 203
  - 75 minute open book exam
  - Review session 3/15; 8PM/room:TBL 202
  - Sample midterm is available online
Today’s Plan

- Recursive classes
public class SkeletalBrowser extends GUIManager {
    ...

    // The complete list of web site addresses that have been entered
    private HistoryList history;
    // Display the elements of the program's user interface
    public SkeletalBrowser() {
        // Create window to hold all the components
        this.createWindow( WINDOW_WIDTH, WINDOW_HEIGHT );
        ...
        history = new HistoryList( );
    }

    // When the user completes the entry of a new address, add it to the
    // history list if it isn't already there
    public void textEntered( ) {
        if ( ! history.contains( AddressEntry.getText() ) ) {
            history = new HistoryList( AddressEntry.getText(), history );
        }
        matches.setText( history.toString() );
    }
    ...
}
public class SkeletalBrowser extends GUIManager {
    ...

    // The complete list of web site addresses that have been entered
    private HistoryList history;

    // Display the elements of the program's user interface
    public SkeletalBrowser() {
        // Create window to hold all the components
        this.createWindow( WINDOW_WIDTH, WINDOW_HEIGHT );
        ...
        history = new HistoryList( );
    }

    // When the user completes the entry of a new address, add it to the
    // history list if it isn't already there
    public void textEntered( ) {
        if ( ! history.contains( AddressEntry.getText() ) ) {
            history = new HistoryList( AddressEntry.getText(), history );
        }
        matches.setText( history.toString() );
        ...
    }
}
public class HistoryList {

private boolean empty = false;     // true if nothing in list
private String firstWebsite;       // The first web site in the list
private HistoryList restOfWebSites;// The rest of the list of web sites

// Create an empty list
public HistoryList( ) {
    empty = true;
}

// Create a larger list from a new website and an existing list
public HistoryList( String newSite, HistoryList existingList ) {
    firstWebsite = newSite;
    restOfWebSites = existingList;
    empty = true;
}

// Produces a single String containing all the entries in the list
// separated by new lines.
public String toString( ) {
    if ( empty ) {
        return ""; // empty list
    } else {
        return firstWebsite + "\n" + ???;
    }
}
}
public class SkeletalBrowser extends GUIManager {
  ...

  // The complete list of web site addresses that have been entered
  private HistoryList history = new HistoryList();

  // Display the elements of the program's user interface
  public SkeletalBrowser() {
    // Create window to hold all the components
    this.createWindow( WINDOW_WIDTH, WINDOW_HEIGHT );
    ...
  }

  // When the user completes the entry of a new address, add it to the
  // history list if it isn't already there
  public void textEntered() {
    if ( ! history.contains( AddressEntry.getText() ) ) {
      history = new HistoryList( AddressEntry.getText(), history );
    }
    matches.setText( history.toString() );
    ...
  }
}
Writing a Recursive Class to represent a collection

- Declare some instance variables
  - a boolean variable named empty
  - variables to represent one member of your collection
  - a “rest” variable of the same type as the class
Writing a Recursive Class to represent a collection

Declare two constructors:

- one that expects no parameters and just sets empty = true;
- One that takes one parameter to determine the value of every instance variable except empty and sets empty = false.
Writing a Recursive Method

- Write some base cases:
  - if ( empty ) {...
  - if ( ! empty ) { ...
  - if ( firstBlah.equals( ... ) ) {...
Writing a Recursive Method

Think about what the result of a recursive call should look like (or what it should do):

- `rest.toString()`
- `rest.contains(name)`
- `rest.setVisible()`
Writing a Recursive Method

“You have to let it all go, Neo. Fear, doubt, and disbelief. Free your mind.”

— Morpheus, The Matrix
Writing a Recursive Method

ASSUME ANY RECURSIVE CALL YOU WRITE WILL WORK CORRECTLY!!!!

Figure out how to determine the correct answer (accomplish the desired result) given the result (effect) of a recursive call.

- first + "\n" + rest.toString()
- first.equals(name) || rest.contains(name)
- first.setVisible();
- rest.setVisible();
public class HistoryList {
    private boolean empty = false; // true if nothing in list
    private String firstWebsite; // The first web site in the list
    private HistoryList restOfWebSites; // The rest of the list of web sites

    ... 

    // determines whether the collection contains a given entry
    public boolean contains( String site ) {
        if ( empty ) {
            ...
        } else {
            ...
        }
    }
}
public class HistoryList {
    private boolean empty = false; // true if nothing in list
    private String firstWebsite; // The first web site in the list
    private HistoryList restOfWebSites; // The rest of the list of web sites

    ...

    // determines whether the collection contains a given entry
    public boolean contains( String site ) {
        if ( empty ) {
            return false;
        } else {
            }
    }
}
}
public class HistoryList {

    private boolean empty = false; // true if nothing in list
    private String firstWebsite; // The first web site in the list
    private HistoryList restOfWebSites; // The rest of the list of web sites

    ...

    // determines whether the collection contains a given entry
    public boolean contains( String site ) {
        if ( empty ) {
            return false;
        } else {
            if ( firstWebsite.equals( site ) ) {
                return true;
            } else {
                ...
            }
        }
    }
}

public class HistoryList {
    private boolean empty = false;  // true if nothing in list
    private String firstWebsite;    // The first web site in the list
    private HistoryList restOfWebSites;   // The rest of the list of web sites

    ...

    // determines whether the collection contains a given entry
    public boolean contains( String site ) {
        if ( empty ) {
            return false;
        } else {
            if ( firstWebsite.equals( site ) ) {
                return true;
            } else {
                return restOfWebSites.contains( site );
            }
        }
    }
}
private boolean empty = false; // true if nothing in list
private String firstWebsite; // The first web site in the list
private HistoryList restOfWebSites; // The rest of the list of web sites

// Create an empty list
public HistoryList( ) {
    empty = true;
}

// Create a larger list from a new website and an existing list
public HistoryList( String newSite, HistoryList existingList ) {
    firstWebsite = newSite;
    restOfWebSites = existingList;
}

// determines whether the collection contains a given entry
public boolean contains( String site ) {
    if ( empty ) {
        return false;
    } else if ( firstWebsite.equals( site ) ) {
        return true;
    } else {
        return restOfWebSites.contains( site );
    }
}