Module 7 Lab 7A

Name: Clayton Black
Date: 11-21-2019
Assignment Name: Module 7 Lab 7B
Assignment Brief: Linked List Sources:

- https://ccse.kennesaw.edu/fye/pseudocode/pseudocodeguide.php

Main

```
CLASS Main
BEGIN

    METHOD Main
    BEGIN
        high ← 0
        low ← 1
        CREATE int[][] calendar ← int[12][2]
        monthCount ← 1

    FOR mon IN calendar
        mon[high] ← getIntInput(String.format("Please Enter Month %d's High: ", monthCount))
        mon[low] ← getIntInput(String.format("Please Enter Month %d's Low: ", monthCount++))
    ENDFOR

    PRINT "Original Lows\n"
    printCal(calendar, low)
    PRINT "Original Highs\n"
    printCal(calendar, high)


    PRINT "Sorted Lows\n"
    sort(calendar, low)
    printCal(calendar, low)


    PRITN "\n\nSorted Highs"
    sort(calendar, high, true);
    printCal(calendar, high);

END METHOD

METHOD getIntInput(prompt)
BEGIN
    int ret
    Scanner scanner ← new Scanner(System.in)
    PRINT prompt
    TRY
        ret ← scanner.nextInt()
    CATCH (Exception e)
        PRINT "Input must be an Integer!"
        ret ← getIntInput(prompt)
    END TRY
END METHOD

METHOD printCal(calendar, index)
BEGIN
    FOREACH mon in calendar
        PRINT mon[index] + " "
    ENDFOREACH
END METHOD

METHOD sort(calendar, index)
BEGIN
    sort(calendar, index, false)
END METHOD

METHOD sort(calendar, index, reverse)
BEGIN
    IF calendar.length > 1 THEN

        halfOne ← int[calendar.length/2][2]
```

```
            arraycopy(calendar, 0, halfOne, 0, calendar.length/2)
            sort(halfOne, index, reverse)

            halfTwoLength ← calendar.length - halfOne.length
            halfTwo ← int[halfTwoLength][2]
            arraycopy(calendar, calendar.length/2, halfTwo, 0, halfTwoLength)
            sort(halfTwo, index, reverse)

            IF reverse THEN
                reverseMerge(halfOne, halfTwo, calendar, index)
            else
                merge(halfOne, halfTwo, calendar, index)
            ENDIF

        ENDIF
    END METHOD

    METHOD merge(list1, list2, temp, index)
    BEGIN

        c1 ← c2 ← c3 ← 0

        WHILE c1 < list1.length && c2 < list2.length THEN
            IF list1[c1][index] < list2[c2][index] THEN
                temp[c3++] ← list1[c1++]
            ELSE
                temp[c3++] ← list2[c2++]
        ENDWHILE

        WHILE c1 < list1.length THEN
            temp[c3++] ← list1[c1++]
        ENDWHILE

        WHILE c2 < list2.length THEN
            temp[c3++] ← list2[c2++]
        ENDWHILE

    END METHOD

    METHOD reverseMerge(list1, list2, temp, index)
    BEGIN

        c1 ← c2 ← c3 ← 0

        WHILE c1 < list1.length && c2 < list2.length THEN
            IF list1[c1][index] > list2[c2][index] THEN
                temp[c3++] ← list1[c1++]
            ELSE
                temp[c3++] ← list2[c2++]
        ENDWHILE

        WHILE c1 < list1.length THEN
            temp[c3++] ← list1[c1++]
        ENDWHILE

        WHILE c2 < list2.length THEN
            temp[c3++] ← list2[c2++]
        ENDWHILE

    END METHOD


END CLASS
```