

Solutions to ISE 208 Final Exam Practice Problems

1. Linked Lists (40 points)

Define a `reverse()` method for the `CSE114List` class. This method should reverse the order of the nodes in the list so that the last node in the old list becomes the first node in the new list, the next-to-last node in the old list becomes the second node in the new list, and so forth. Your method should ultimately change the original list to its reversed version. You may add any helper methods that you need.

For example, if the original order of nodes was "apple", "banana", "grape", after calling `reverse()`, the list would contain "grape", "banana", "apple", in that order.

Sample method header: `public void reverse ()`

```
// The simplest solution is to create a second, temporary list. Extract
// the elements of the original list into the temporary list, and then
// rebuild the original list in the correct order
public void reverse ()
{
    if (head == null) // don't do anything to an empty list
    {
        return;
    }

    CSE114List temp = new CSE114List(); // create the second list
    while (head != null) // Move everything out of the original list
    {
        String x = removeFromEnd();
        temp.insertAtEnd(x); // build the new list in reverse order
    }

    // Transfer everything back to the original list
    while (temp.size() > 0)
    {
        String w = temp.removeFromEnd();
        insertAtFront(w);
    }
}
```

2. Arrays (30 points)

Add a method named `interleave()` to the `ArrayProblem` class. This method takes two integer arrays as its arguments and combines their contents in the following order:

1. The odd values in the first array, in the order they appear in that array
2. The odd values in the second array, in reverse order
3. The even values in the second array, stored in the order they appear in that array
4. The even values in the first array, stored in reverse order

Your method should return the resulting array. You **MAY NOT** assume that the input arrays are of equal size!

Sample header: `public static int [] interleave (int [] first, int [] second)`

```
public static int [] interleave (int [] first, int [] second)
```

```
{
```

```
    int [] temp = new int [first.length + second.length];
    int currentPosition = 0; // location in the temp array
```

```
    // Part 1
```

```
    for (int i = 0; i < first.length; i++)
```

```
    {
```

```
        if (first[i] % 2 == 1) // odd value
```

```
        {
```

```
            temp[currentPosition] = first[i];
```

```
            currentPosition++;
```

```
        }
```

```
    }
```

```
    // Part 2
```

```
    for (int j = second.length - 1; j >= 0; j--)
```

```
    {
```

```
        if (second[j] % 2 == 1) // odd value
```

```
        {
```

```
            temp[currentPosition] = second[j];
```

```
            currentPosition++;
```

```
        }
```

```
    }
```

```
// Part 3
for (int k = 0; k < second.length; k++)
{
    if (second[k] %2 == 0) // even value
    {
        temp[currentPosition] = second[k];
        currentPosition++;
    }
}

// Part 4
for (int m = first.length - 1; m >= 0; m --)
{
    if (first[m] % 2 == 0) // even value
    {
        temp[currentPosition] = first[m];
        currentPosition++;
    }
}

return temp;
}
```

3. General Programming (30 points)

Complete the code for the Balancer program.

```
public static boolean isBalanced (String input)
{
    int numAs = 0;
    int numBs = 0;
    int numCs = 0;

    for (int i = 0; i < input.length(); i++)
    {
        char next = input.charAt(i);
        if (c == 'a')
        {
            numAs++;

            // Check to see if this 'a' follows a 'b' or a 'c'
            if ( (numBs + numCs) > 0)
            {
                return false; // This string is not well-formed
            }
        }
        else if (next == 'b')
        {
            numBs++;

            // Check to make sure we haven't seen any c's yet
            if (numCs > 0)
            {
                return false; // This string is not well-formed
            }
        }
        else
        {
            numCs++;
        }
    }
}
```

```
    // Check for the proper ratio of a's, b's, and c's
    if ( (numAs + numBs) == numCs)
    {
        return true;
    }
    else
    {
        return false;
    }
} // end of for loop
}
```