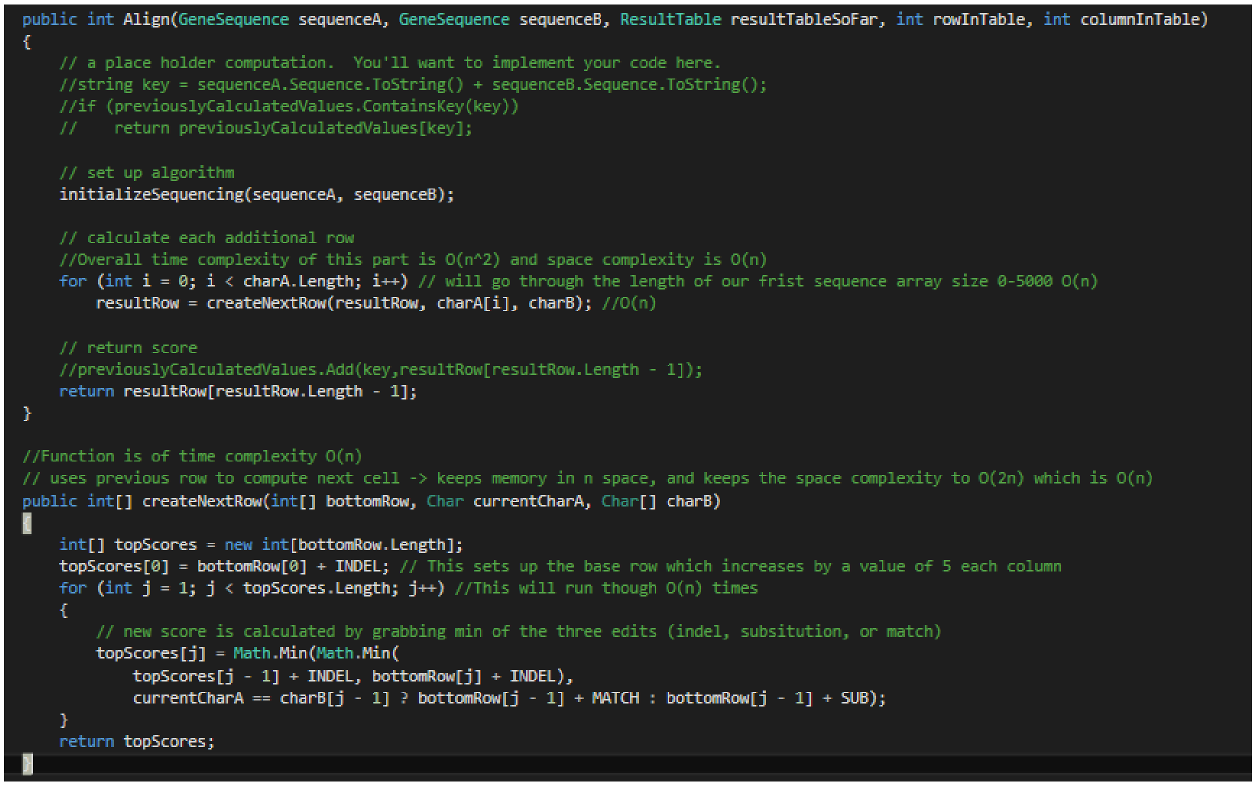
Anthony Constantino
8 November 2015
Project 4 Report

Gene Sequencing

Scoring Algorithm

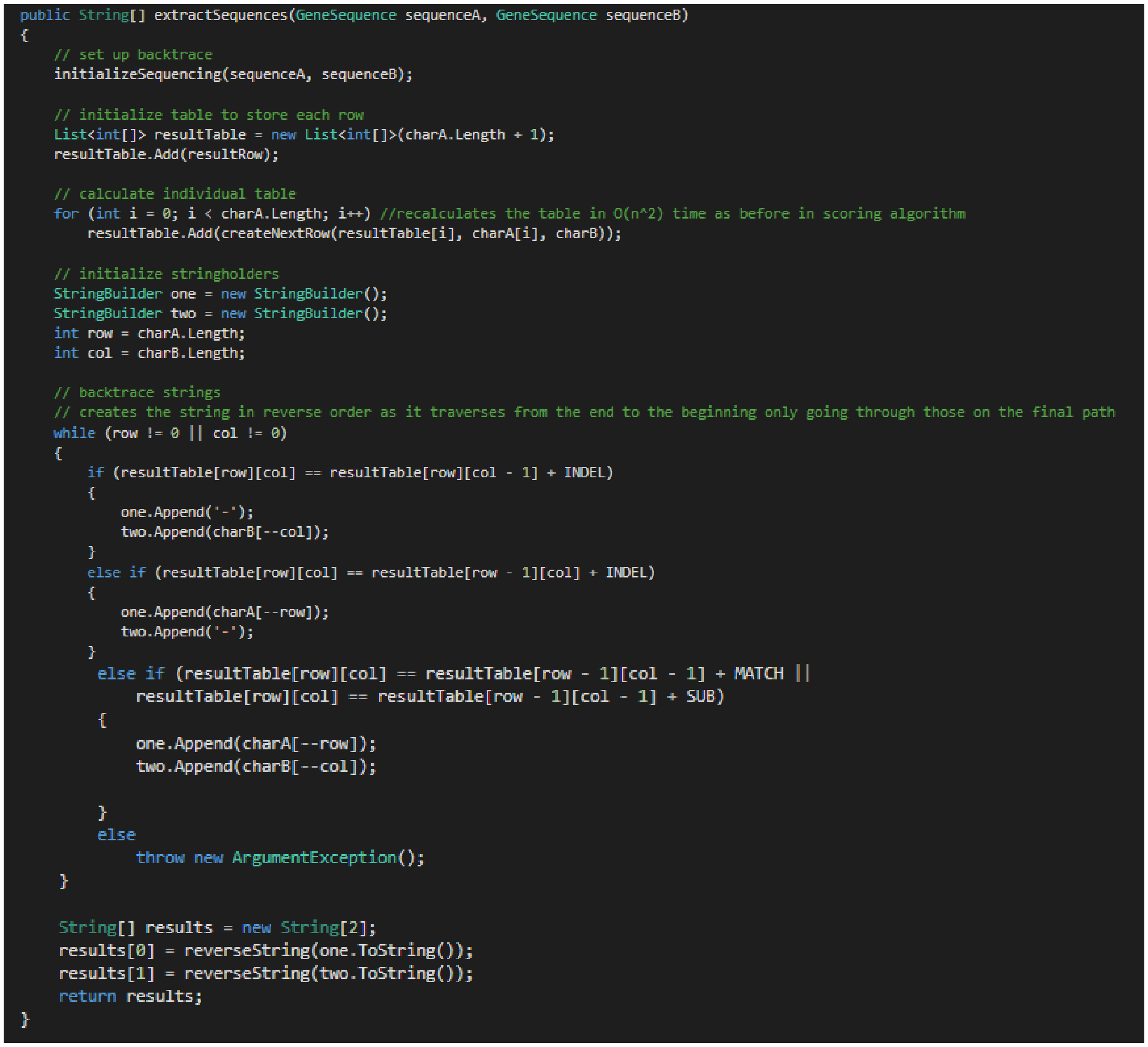
In the scoring algorithm runs through two rows at a time to keep the space complexity down to O(n). The program will then go through starting with the top two rows and begin calculating distance values. Running through the length of the first sequence and then the length of the second making the complexity O(n^2)



```
public int Align(GeneSequence sequenceA, GeneSequence sequenceB, ResultTable resultTableSoFar, int rowInTable, int columnInTable)
{
    // a place holder computation.  You'll want to implement your code here.
    //string key = sequenceA.Sequence.ToString() + sequenceB.Sequence.ToString();
    //if (previouslyCalculatedValues.ContainsKey(key))
    //    return previouslyCalculatedValues[key];

    // set up algorithm
    initializeSequencing(sequenceA, sequenceB);

    // calculate each additional row
    //Overall time complexity of this part is O(n^2) and space complexity is O(n)
    for (int i = 0; i < charA.Length; i++) // will go through the length of our frist sequence array size 0-5000 O(n)
        resultRow = createNextRow(resultRow, charA[i], charB); //O(n)

    // return score
    //previouslyCalculatedValues.Add(key,resultRow[resultRow.Length - 1]);
    return resultRow[resultRow.Length - 1];
}

//Function is of time complexity O(n)
// uses previous row to compute next cell -> keeps memory in n space, and keeps the space complexity to O(2n) which is O(n)
public int[] createNextRow(int[] bottomRow, Char currentCharA, Char[] charB)
{
    int[] topScores = new int[bottomRow.Length];
    topScores[0] = bottomRow[0] + INDEL; // This sets up the base row which increases by a value of 5 each column
    for (int j = 1; j < topScores.Length; j++) //This will run though O(n) times
    {
        // new score is calculated by grabbing min of the three edits (indel, subsitution, or match)
        topScores[j] = Math.Min(Math.Min(
            topScores[j - 1] + INDEL, bottomRow[j] + INDEL),
            currentCharA == charB[j - 1] ? bottomRow[j - 1] + MATCH : bottomRow[j - 1] + SUB);
    }
    return topScores;
}
```

Extraction Algorithm

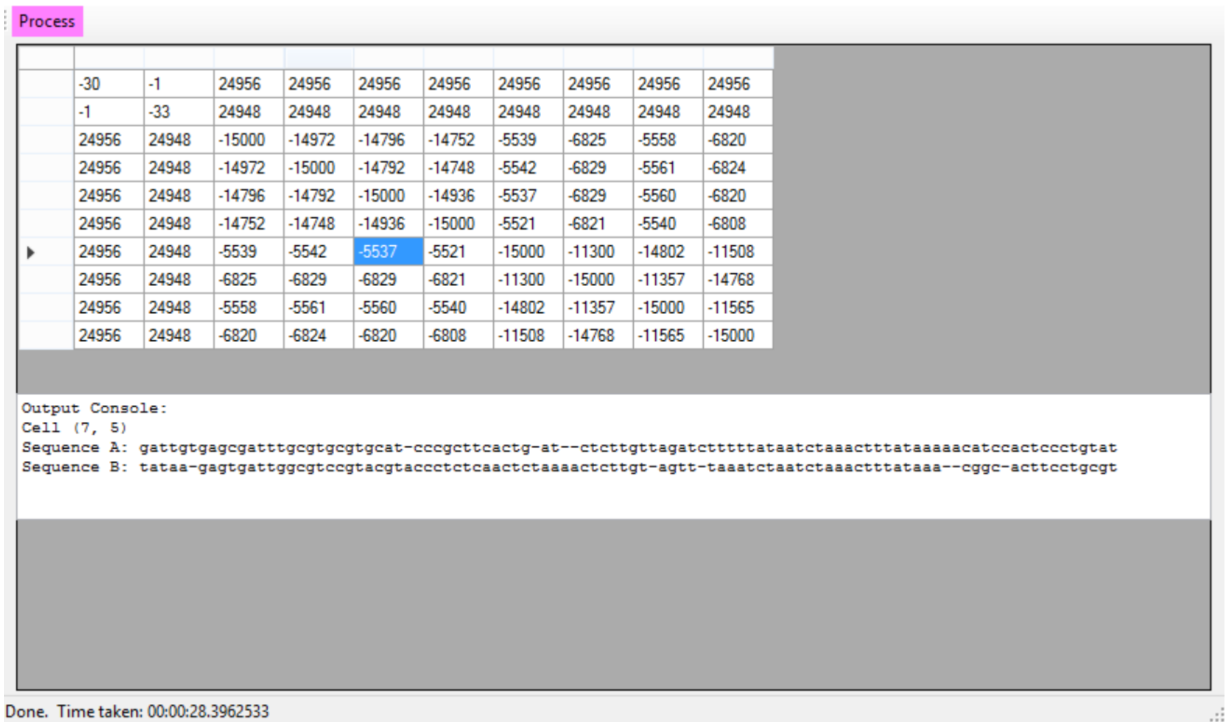
In the extraction algorithm the way I grabbed the results was fairly similar to the scoring algorithm. I started by going through the table and going through calculating the shortest distance between the two sequences. Once the algorithm has calculated the path to the bottom right value of the table. Then as the algorithm reached the end it comes back up, it generates the two words from the movements that were made to reach the bottom. Once the program has finished the loop the string is then reversed to present the resulting match up.



```
public String[] extractSequences(GeneSequence sequenceA, GeneSequence sequenceB)
{
    // set up backtrace
    initializeSequencing(sequenceA, sequenceB);

    // initialize table to store each row
    List<int[]> resultTable = new List<int[]>(charA.Length + 1);
    resultTable.Add(resultRow);

    // calculate individual table
    for (int i = 0; i < charA.Length; i++) //recalculates the table in O(n^2) time as before in scoring algorithm
        resultTable.Add(createNextRow(resultTable[i], charA[i], charB));

    // initialize stringholders
    StringBuilder one = new StringBuilder();
    StringBuilder two = new StringBuilder();
    int row = charA.Length;
    int col = charB.Length;

    // backtrace strings
    // creates the string in reverse order as it traverses from the end to the beginning only going through those on the final path
    while (row != 0 || col != 0)
    {
        if (resultTable[row][col] == resultTable[row][col - 1] + INDEL)
        {
            one.Append('-');
            two.Append(charB[--col]);
        }
        else if (resultTable[row][col] == resultTable[row - 1][col] + INDEL)
        {
            one.Append(charA[--row]);
            two.Append('-');
        }
         else if (resultTable[row][col] == resultTable[row - 1][col - 1] + MATCH ||
             resultTable[row][col] == resultTable[row - 1][col - 1] + SUB)
         {
             one.Append(charA[--row]);
             two.Append(charB[--col]);

         }
         else
             throw new ArgumentException();
     }

     String[] results = new String[2];
     results[0] = reverseString(one.ToString());
     results[1] = reverseString(two.ToString());
     return results;
}
```

Results

10 by 10 picture with time



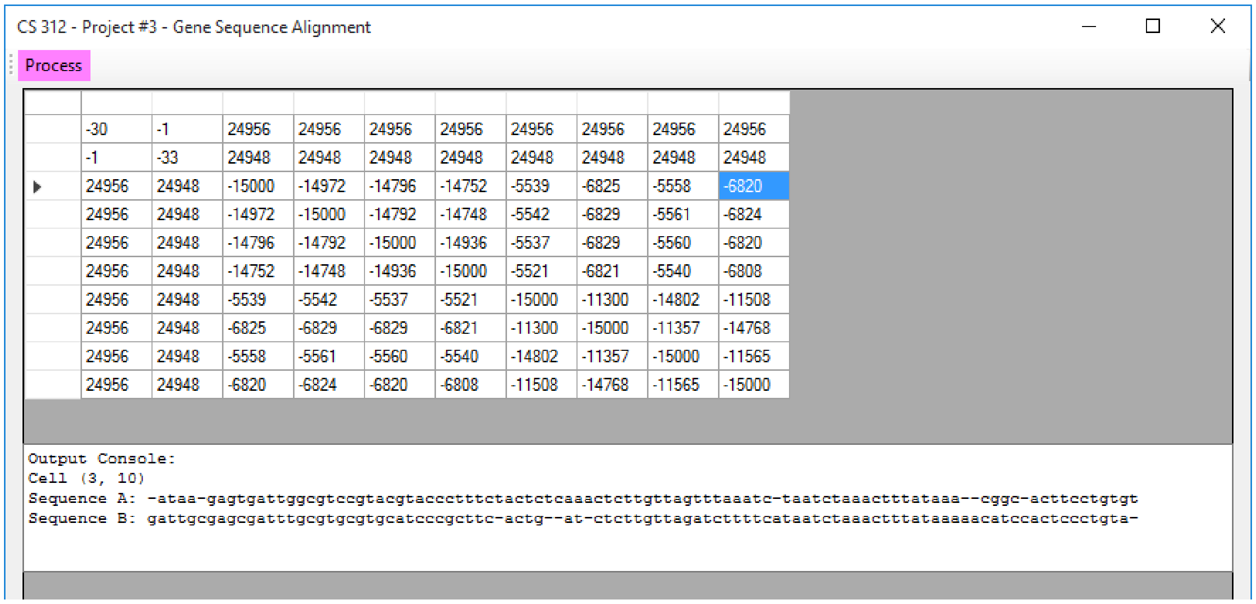
Process

-30	-1	24956	24956	24956	24956	24956	24956	24956	24956
-1	-33	24948	24948	24948	24948	24948	24948	24948	24948
24956	24948	-15000	-14972	-14796	-14752	-5539	-6825	-5558	-6820
24956	24948	-14972	-15000	-14792	-14748	-5542	-6829	-5561	-6824
24956	24948	-14796	-14792	-15000	-14936	-5537	-6829	-5560	-6820
24956	24948	-14752	-14748	-14936	-15000	-5521	-6821	-5540	-6808
24956	24948	-5539	-5542	-5537	-5521	-15000	-11300	-14802	-11508
24956	24948	-6825	-6829	-6829	-6821	-11300	-15000	-11357	-14768
24956	24948	-5558	-5561	-5560	-5540	-14802	-11357	-15000	-11565
24956	24948	-6820	-6824	-6820	-6808	-11508	-14768	-11565	-15000

```
Output Console:
Cell (7, 5)
Sequence A: gattgtgagcgatttgcgtgcgtgcat-cccgcttcactg-at--ctcttgttagatcttttataatctaactttataaaacatccactccctgtat
Sequence B: tataa-gagtgattggcgtccgtacgtaccctctcaactctaaaactcttgt-agtt-taaatctaatctaactttataaa--cggc-acttcctgcgt
```

Done. Time taken: 00:00:28.3962533

Side by side comparison of 3,10



CS 312 - Project #3 - Gene Sequence Alignment

Process

-30	-1	24956	24956	24956	24956	24956	24956	24956	24956
-1	-33	24948	24948	24948	24948	24948	24948	24948	24948
24956	24948	-15000	-14972	-14796	-14752	-5539	-6825	-5558	-6820
24956	24948	-14972	-15000	-14792	-14748	-5542	-6829	-5561	-6824
24956	24948	-14796	-14792	-15000	-14936	-5537	-6829	-5560	-6820
24956	24948	-14752	-14748	-14936	-15000	-5521	-6821	-5540	-6808
24956	24948	-5539	-5542	-5537	-5521	-15000	-11300	-14802	-11508
24956	24948	-6825	-6829	-6829	-6821	-11300	-15000	-11357	-14768
24956	24948	-5558	-5561	-5560	-5540	-14802	-11357	-15000	-11565
24956	24948	-6820	-6824	-6820	-6808	-11508	-14768	-11565	-15000

```
Output Console:
Cell (3, 10)
Sequence A: -ataa-gagtgattggcgtccgtacgtaccctttctactctcaaactcttgttagtttaaatc-taatctaactttataaa--cggc-acttcctgtgt
Sequence B: gattgcgagcgatttgcgtgcgtgcatcccgcttc-actg--at-ctcttgttagatcttttcataatctaactttataaaacatccactccctgta-
```