CSC505 Jennings
Homework 5, Spring 2020 SAMPLE SOLUTION
Please implement the programs and answer the 9 questions.

This assignment may be done alone or in groups of 2 or 3 people. Turn in one copy of the written part of the assignment on Gradescope, and enter the names of each team member during the submission process.

I. Overview:

Do not discuss this homework with other teams.

In this assignment, you will implement a version of the popular `diff` program using (1) a string hash function and (2) an algorithm for the Longest Common Subsequence problem, using the Dynamic Programming paradigm.

For this assignment, you can turn in a PDF in almost any format – there is no template to follow. On Gradescope, you will have to indicate where in your PDF we will find the answer to each question.

You must provide (via NCSU github) to the teaching staff any source code you use and scripts that can be used to build any code and run any/all steps needed to reproduce your results.

And, as always, you must cite the origin of all source code, which may be original to your team or individual team members, or may be appropriately obtained from elsewhere.

These programming languages are acceptable for source code used in this assignment:

Java 12
Python 3.7
C (C11 or ANSI)
C++ 17

Checklist:

- ☐ Read and answer the questions on the following pages.
- ☐ Implement from scratch OR use code found online or in a library (which you must cite) the algorithms needed for this assignment.
- ☐ Tell Gradescope who worked on your team assignment. (One, two, or three people may work on it.)
- ☐ Do not forget to cite all resources you used, including our textbook.
- ☐ Add the teaching staff to your NCSU github repository.
- ☐ Be sure to include a link to your NCSU github repository.

II. Implementation

A. Implement a hash function for strings. Use a *modulo* function to compress the resulting value to 10 bits, i.e. to be in the range [0, 1023].
For example:

```
hash10("Hello!") ➔ 429
hash10("") ➔ 0
hash10("\n\n\n") ➔ 74
```

Your compressed hash values may differ.

B. Use your hash function to implement a `hashfile` program that takes a filename on the command line and produces a 10-bit hash of each line.
For example:

```
$ hashfile f1.txt
517
567
263
$ hashfile f2.txt
517
263
76
765
```

You can see that f1.txt has 3 lines, and f2.txt has 5 lines. The hash values printed by your program will probably be different.

If your program requires an interpreter, like python3, then make a bash script that runs your program using the interpreter. The teaching staff must be able to invoke your program as shown.

C. Implement the dynamic programming algorithm for `diff`, using the dynamic programming approach to the Longest Common Subsequence problem. Use the code from your `hashfile` implementation. (You do not need to run `hashfile` and use its output – just be sure to re-use your hashing and file i/o code.) The output of `diff` should be two lists of line numbers, one for file1 and one for file2. The lists are the same length and contain the line numbers *(with 1 as the lowest line number)* that are the same in the two files.

For example, we can see by the hash numbers in the previous example that line 1 is the same in f1.txt and f2.txt (they hash to 517); and line 3 in f1.txt is the same as line 2 of f2.txt (they hash to 263). All other lines are different.

Sample output from `diff`:

```
$ diff f1.txt f2.txt
1 3
1 2
```

You can check your program using the sample files provided at https://github.ncsu.edu/jajenni3/csc505/tree/master/HW5.

D. FOR BONUS POINTS:
Implement a program `diffprint` that uses the code you wrote for `diff` and prints a simplified form of the "standard" diff output showing the contents of both files. Lines that are common to both files are printed with two spaces in front of them. Lines present only in the second file are printed with "+ " (plus, then space) in front, and lines present only in the first file are printed with "- " (minus, then space) in front.

For example:

```
$ diffprint f1.txt f2.txt
  Hello
- there
  world.
+ Really!
+ Yep.
```

III. Questions:

1. Names of the people in the team:
   (without this, you will receive 0 for the assignment)

   _____

   _____

   _____

2. NCSU github URL: (without this, you will receive 0 for the assignment)

   https://github.ncsu.edu/jajenni3/csc505/tree/master/DynamicProgramming/diff

3. Citations for code: (without this, you will receive 0 for the assignment)
   Please cite non-code resources in your answers to specific questions.

   _____ (Use more space as needed.)

   _____

   _____

4. What is the output of your `diff` program on files ex41.txt, ex42.txt?

Show the command line invocation and the output

Observe that the two output lines are the same in each case, but they appear in reverse order when the arguments are in reverse order:

```
$ ./diff ex41.txt ex42.txt
1 2 3 4 5 6 10 11 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66
2 3 4 5 6 7 10 12 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41
$ ./diff ex42.txt ex41.txt
2 3 4 5 6 7 10 12 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41
1 2 3 4 5 6 10 11 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56 58 60 62 64 66
$
```

5. What is the output of your `diff` program on files ex42.txt, ex41.txt?

Show the command line invocation and the output

See above.

Else leave blank

6. If you implemented **`diffprint`**, give the output here…
    a. For files ex61.txt, ex62.txt

```
$ ./diffprint ex61.txt ex62.txt
  CONTENTS

- CHAPTER I                                                    Page
+                                                             Page
  Jonathan Harker's Journal                                      1
- CHAPTER II
  Jonathan Harker's Journal                                     14
- CHAPTER III
  Jonathan Harker's Journal                                     26
- CHAPTER IV
  Jonathan Harker's Journal                                     38
- CHAPTER V
  Letters--Lucy and Mina                                        51
- CHAPTER VI
  Mina Murray's Journal                                         59
- CHAPTER VII
  Cutting from "The Dailygraph," 8 August                       71
- CHAPTER VIII
  Mina Murray's Journal                                         84
- CHAPTER IX
  Mina Murray's Journal                                         98
- CHAPTER X
  Mina Murray's Journal                                        111
- CHAPTER XI
  Lucy Westenra's Diary                                        124
- CHAPTER XII
  Dr. Seward's Diary                                           136
- CHAPTER XIII
  Dr. Seward's Diary                                           152
- CHAPTER XIV
  Mina Harker's Journal                                        167
- CHAPTER XV
  Dr. Seward's Diary                                           181
- CHAPTER XVI
  Dr. Seward's Diary                                           194
- CHAPTER XVII
  Dr. Seward's Diary                                           204
- CHAPTER XVIII
  Dr. Seward's Diary                                           216
- CHAPTER XIX
  Jonathan Harker's Journal                                    231
- CHAPTER XX
  Jonathan Harker's Journal                                    243
- CHAPTER XXI
  Dr. Seward's Diary                                           256
- CHAPTER XXII
  Jonathan Harker's Journal                                    269
- CHAPTER XXIII
  Dr. Seward's Diary                                           281
- CHAPTER XXIV
  Dr. Seward's Phonograph Diary, spoken by Van Helsing         294
- CHAPTER XXV
  Dr. Seward's Diary                                           308
- CHAPTER XXVI
  Dr. Seward's Diary                                           322
- CHAPTER XXVII
  Mina Harker's Journal                                        338
```

b. For files ex62.txt, ex61.txt

```
$ ./diffprint ex62.txt ex61.txt
  CONTENTS

-                                                                    Page
+ CHAPTER I                                                          Page
  Jonathan Harker's Journal                                             1
+ CHAPTER II
  Jonathan Harker's Journal                                            14
+ CHAPTER III
  Jonathan Harker's Journal                                            26
+ CHAPTER IV
  Jonathan Harker's Journal                                            38
+ CHAPTER V
  Letters--Lucy and Mina                                               51
+ CHAPTER VI
  Mina Murray's Journal                                                59
+ CHAPTER VII
  Cutting from "The Dailygraph," 8 August                              71
+ CHAPTER VIII
  Mina Murray's Journal                                                84
+ CHAPTER IX
  Mina Murray's Journal                                                98
+ CHAPTER X
  Mina Murray's Journal                                               111
+ CHAPTER XI
  Lucy Westenra's Diary                                               124
+ CHAPTER XII
  Dr. Seward's Diary                                                  136
+ CHAPTER XIII
  Dr. Seward's Diary                                                  152
+ CHAPTER XIV
  Mina Harker's Journal                                               167
+ CHAPTER XV
  Dr. Seward's Diary                                                  181
+ CHAPTER XVI
  Dr. Seward's Diary                                                  194
+ CHAPTER XVII
  Dr. Seward's Diary                                                  204
+ CHAPTER XVIII
  Dr. Seward's Diary                                                  216
+ CHAPTER XIX
  Jonathan Harker's Journal                                           231
+ CHAPTER XX
  Jonathan Harker's Journal                                           243
+ CHAPTER XXI
  Dr. Seward's Diary                                                  256
+ CHAPTER XXII
  Jonathan Harker's Journal                                           269
+ CHAPTER XXIII
  Dr. Seward's Diary                                                  281
+ CHAPTER XXIV
  Dr. Seward's Phonograph Diary, spoken by Van Helsing                294
+ CHAPTER XXV
  Dr. Seward's Diary                                                  308
+ CHAPTER XXVI
  Dr. Seward's Diary                                                  322
+ CHAPTER XXVII
  Mina Harker's Journal                                               338
```

c. For files ex66a.txt, ex66b.txt

Note the blank line at the start. It is a part of the LCS, i.e. both files begin with a blank line.

```
$ ./diffprint ex66a.txt ex66b.txt

- A
- B
+ A
+ B
  C
  D
  E
  F
  G
- H
+ H
+ I
+ J
$
```

7. Your implementation of LCS requires how much space? Your answer should use big-Theta, because your algorithm cannot use less and will never need more.

$\Theta(nm)$ where n is the number of lines in file 1 and m is the number of lines in file 2. Generally, we would say these are the number of elements of the first sequence and of the second sequence, respectively.

A short answer is appropriate. Be sure to explain the meaning of any variables like m or n.

8. How do you know that the 10-bit hash code required by this assignment is sufficient to allow your `diff` program to work correctly? (How would you know if it was not sufficient?)

A few sentences should suffice.

We cannot know that 10 bits is sufficient simply by *using* our diff solution. If we always validate the correctness of the output, then I would say we are *testing* our diff solution, and in that case we may find errors in the output.
An error would be due to a hash collision, which would cause diff to believe that two different lines were identical.
The result would be a longer LCS (produced in error) than the actual LCS.
It is acceptable to say that we know 10 bits is insufficient if we can show that it produces hash collisions. An alternative answer to that part of the question is to say that we can know if 10 bits is insufficient by testing thoroughly against generated or actual data to look for collisions, in the same way that hash functions are tested for use in hash tables (to see how often they produce collisions on "likely data", like English words).

9. Suppose you were tasked with writing a `diff` program that would work reliably on input files that were long (up to, e.g. 10,000 lines) and could have long lines (up to, e.g. 5,000 bytes).

a. What are the implications for the choice of hash function? (Describe what properties you would want in a hash function.)

There are several points you could potentially make here.

The long lines and large quantity of lines increases the probability that our hash function may generate colliding values, which will cause an error in our diff output. We want the best hash function we can have for the kind of data that will be hashed – meaning, one that generates collisions with extremely low probability.
Cryptographic hashes have this property, by the way. It is not necessary to state this in your answer, but hash functions like SHA-256 and others have very low probability of producing collisions when used correctly. Such hash functions are much slower than the hash functions typically used in hash tables (where collisions affect only performance, not correctness).

b. What are the implications for actual memory usage? I.e. calculate a reasonable approximation of how much memory your program might need in the worst case.

First identify the main consumer(s) of memory that is needed during processing of LCS.

Actual memory usage will be essentially independent of the line lengths. It is likely that any hash function can be computed incrementally by examining one byte or a chunk of bytes at a time from a file. The amount of memory needed while hashing is then some fixed amount, making the line length not relevant.

However, the number of lines in the input files directly affect the size of the matrix needed to calculate the LCS. The example given was 10,000, which is rather low. If we had 2 files of this size, we would have 100×10^6 entries in the matrix, not counting the null row and column. How big is each entry? It stores a count that can go as high as the length of the LCS, which is, in this example, only 10,000. Numbers as high as 16383 can be stored in 16 bits, so if our files were never longer than that, we would need space for 100×10^6 entries at 2 bytes each, or 200×10^6 bytes, which is about 191Mb.
(For files up to about 4Gb in size, we would need double that storage because we would need 32 bits to store the line number of a file.)

Note that hash storage is much less. If we used the hash function from this assignment, then we need to store 10 bits per hash, or 20,000 × 10 = 200,000 bits. But we would typically use a 16-bit int, so we would need 20,000 × 2 = 40,000 bytes, or about 39Kb. And if we used 64-bit (8 byte) hash values, we would need 20,000 × 8 bytes = 160,000 bytes, or about 156Kb.