Assignment 1

z5131048 Fiona Lin

August 15, 2018

1 Task 1

The program that mimics uniq filters adjacent matching lines from INPUT (or standard input), writing to OUTPUT (or standard output). It is assuming the input has an array of strings a and a variable n contains a non-negative integer, the number of elements of the array a, and the output has an array of strings b stores k (a non-negative integer) elements same in the array a but having any adjacent identical strings removed. The program should changes neither n nor a.

With a non-negative integer n, the array a contains n strings elements, the array a in fact is a 2-dimesion char array. As an element within the array a, the string with arbitary length ℓ contains a permutation of characters, based on certain ASCII encoding. Having defined that string $a[i]$ with a length ℓ with 8-bit ASCII encoding by

$$\mathsf{str}(a, n) = \forall i \in 0..(n-1)\,(\wedge)\,\exists \ell \in N \wedge \forall h \in 0..(\ell - 1)\,(\exists e \in 0..255\,(a[i][h] = e))$$

It is defined the array a contains n strings by: $\forall i \in 0..(n-1)\,(str(a, n))$

The precondition is defined as:

$$\forall i \in 0..(n-1)\,(str(a, n) \wedge a) = a_0$$

Thus it states that the array a contains n string elements, and it has captured the initial value for later reference in the postcondition

$$\forall i \in 0..(k-2)\,(str(b, k) \subset str(a, n) \wedge a = a_0 \wedge (b[i] \neq b[i+1]))$$

Having assumed that the output has an array of strings b stores k (a non-negative integer) elements same in the array a but having any adjacent identical strings removed. The program should leave the integer n and the array a remaining the same as original.

2 Task 2

We propose the following proof outline to demonstrate the correctness of our code (in black).

$\{str(a,n) \wedge a = a_0\}$
$i := 0;$ $\quad 1st \Rightarrow \{I[^0/_i][^0/_k]\}$
$k := 0;$
$\{I\}$
while $i < n$ **do** $\quad \{I \wedge i < n\}$
if $i = 0$
$\{I \wedge i = 0\}$ $\quad 2nd \Rightarrow$ $\quad \{I[^0/_i][^0/_k][^{(b:k \mapsto a[i])}/_b]\}$
$b[k] := a[i];$
else
$\{I[^{i+1}/_i][^{k+1}/_k] \wedge 1 \leq i < n\}$
$h := 0;$ $\quad \{J[^0/_h]\}$
$\{J\}$
while $a[i][h] \neq' \backslash 0' \wedge b[k][h] = a[i][h]$ **do**
$\{J \wedge 0 \leq h \wedge a[i][h] \neq' \backslash 0' \wedge b[k][h] = a[i][h]\}$
$\{J[^{h+1}/_h]\}$
$h := h + 1;$
$\{J\}$
od
$\{J \wedge (\forall h \in N\,(b[k][h] = a[i][h])) \vee \exists h \in N\,(b[k][h] \neq a[i][h]))\}$
$5th \Rightarrow \{I[^{i+1}/_i][^{k+1}/_k] \wedge (\forall h \in N\,(b[k][h] = a[i][h])) \vee \exists h \in N\,(b[k][h] \neq a[i][h]))\}$
if $a[i][h] \neq b[k][h]$
$\{(0 < i < n) \wedge \exists h \in N\,(a[i][h] \neq b[k][h]) \wedge I[^{i+1}/_i]\}$
$k := k + 1$ $\quad 3rd \Rightarrow \{I[^{i+1}/_i][^{k+1}/_k][^{(b:k \mapsto a[i])}/_b]\}$
$b[k] := a[i];$
fi
fi
$\{I[^{i+1}/_i]\}$
$i := i + 1$
$\{I\}$
od;
$\{I \wedge i \geq n\}$ $\quad 4th \Rightarrow$
$\{\forall i \in 0..(k-2)\,(str(b,k) \subset str(a,n) \wedge a = a_0 \wedge (b[i] \neq b[i+1]))\}$

where our invariants are

$$I = str(a_0, n) \wedge a = a_0 \wedge (0 \leq i < n) \wedge (0 \leq j < k - 2)$$
$$\wedge \forall j \in 0..(k-2)\,(str(b,k) \subset str(a,i) \wedge (b[j] \neq b[j+1]))$$
$$J = I[^{i+1}/_i][^{k+1}/_k] \wedge i \geq 1 \wedge (\exists h \in N\,(b[k][h] \neq a[i][h]) \vee \forall h \in N\,(a[i][h] = b[k][h]))$$

2.1 First implication: $str(a, n) \wedge a = a_0 \Rightarrow I[^0/_i][^0/_k]$

$\mathsf{str}(a, n) \wedge a = a_0$
$\Rightarrow$ $\langle$using $n \in \mathbb{N}$ and realising that the last conjunct is vacuously true$\rangle$

$\mathsf{str}(a_0, n) \wedge a = a_0 \wedge (0 \leq i < n) \wedge (0 \leq j < k - 2) \wedge$
$\forall j \in 0..(k-2)\,(str(b,k) \subset str(a,i) \wedge (b[j] \neq b[j+1]))$
$\Rightarrow$ $\langle$definitions of I and substitution, since k is 0, str(b,k) is str(b,0)$\rangle$
$\mathsf{str}(a_0, n) \wedge a = a_0 \wedge (0 \leq 0 < n) \wedge (0 \leq j < 0) \wedge$
$\forall j \in 0..(0-2)\,(str(b,0) \subset str(a,0) \wedge (b[j] \neq b[j+1]))$
$\Leftrightarrow$ $\langle$definitions of I and substitution$\rangle$
$I[^0/_i][^0/_k]$

Since str(b,0) is $\emptyset$ similarly str(a,0) is $\emptyset$ hence $str(b,0) \subset str(a,0)$ and $b[0+1]$ is non-exist, then $b[1] \neq b[0]$ is always true.

2.2 Second implication: $I \wedge i = 0 \Rightarrow I[^0/_i][^0/_k][^{(b:k \mapsto a[i])}/_b]$

$I \wedge i = 0$
$\Rightarrow$ $\langle$definitions of I and substitution$\rangle$

$i = 0 \wedge 0 \leq i < n \wedge str(a_0, n) \wedge a = a_0 \wedge (0 \leq i < n) \wedge (0 \leq j < k - 2) \wedge$
$\forall j \in 0..(k-2)\,(str(b,k) \subset str(a,i) \wedge (b[j] \neq b[j+1]))\,[^0/_i][^0/_k]$
$\Rightarrow$ $\langle$simplify, since i and k is 0, str(b, 0) is same as str(a,0)$\rangle$
$0 \leq 0 < n \wedge str(a_0, n) \wedge a = a_0 \wedge (0 \leq 0 < n) \wedge (0 \leq j < 0 - 2) \wedge$
$\forall j \in 0..(0-2)\,(str(b,0) \subset str(a,0) \wedge (b[j] \neq b[j+1]))\,[^{(b:0 \mapsto a[0])}/_b]$
$\Rightarrow$ $\langle$since str(b,0) is $\emptyset$ and $b[0+1]$ is non-exist, then $b[1] \neq b[0]$ is always true$\rangle$
$0 \leq 0 < n \wedge \mathsf{str}(a_0, n) \wedge a = a_0 \wedge 0 \leq 0 < -2 \wedge$
$\forall j \in 0..(0-2)\,(str(b,0) \subset str(a,n) \wedge (b[0] \neq a[0+1]))$
$\Leftrightarrow$ $\langle$similarly $b[0+1]$ is non-exist, then $b[1] \neq a[0]$ is true$\rangle$
$I[^0/_i][^0/_k][^{(b:k \mapsto a[i])}/_b]$

2.3 Third implication:

$(0 < i < n) \wedge \exists h \in N\,(a[i][h] \neq b[k][h]) \wedge I[^{i+1}/_i] \Rightarrow$
$I[^{i+1}/_i][^{k+1}/_k][^{(b:k\mapsto a[i])}/_b]$

First we expand I and perform the substitutions to arrive at

$I \wedge (0 < i < n) \wedge \exists h \in N\,(a[i][h] \neq b[k][h]) \wedge I[^{i+1}/_i]$
$\Rightarrow$ $\langle$definitions of I and substitution$\rangle$

$0 < i+1 < n \wedge \exists h \in N\,(a[i+1][h] \neq b[k][h]) \wedge str(a_0, n) \wedge a = a_0 \wedge (0 \leq j < k-2)$
$\wedge \forall j \in 0..k-2\,(str(b,k) \subset str(a, i+1) \wedge (b[j] \neq b[j+1]))$
$\Rightarrow$ $\langle$let j = k, since $a[i+1][h] \neq b[k][h]$, $str(b, k+1) \subset str(a, i+1)$ remains true$\rangle$
$0 < i+1 < n \wedge \exists h \in N\,(a[i+1][h] \neq b[k][h]) \wedge str(a_0, n) \wedge a = a_0 \wedge (0 \leq j < k-2)$
$\wedge (\forall j \in 0..(k-2)\,(str(b,k) \subset str(a, i+1) \wedge b[j] \neq b[j+1]) \wedge (j = k \wedge b[k] \neq a[i+1]))$
$\Rightarrow$ $\langle$merger $a[i+1][h] \neq b[k][h]$ to str(b,k)$\rangle$
$0 < i+1 < n \wedge \exists h \in N\,(a[i+1][h] \neq b[k][h]) \wedge str(a_0, n) \wedge a = a_0 \wedge (0 \leq j < k-1)$
$\wedge \forall j \in 0..(k-1)\,(str(b, k+1) \subset str(a, i+1) \wedge (b[j] \neq b[j+1])$
$\Rightarrow$ $\langle$simplify$\rangle$
$0 < i+1 < n \wedge str(a_0, n) \wedge a = a_0 \wedge (0 \leq j < k-1)$
$\wedge \forall j \in 0..k-1\,(str(b, k+1) \subset str(a, i+1) \wedge b[j] \neq b[j+1])$
$\Leftrightarrow$ $\langle$definitions of I and substitution$\rangle$
$I[^{i+1}/_i][^{k+1}/_k][^{(b:k\mapsto a[i])}/_b]$

2.4 Forth implication: $I \wedge i \geq n \Rightarrow \forall i \in$ $0..(k-2)\,(str(b,k) \subset str(a,n) \wedge a = a_0 \wedge (b[i] \neq b[i+1]))$

$I \wedge i \geq n$
$\Rightarrow$ $\langle$definitions of I and substitution$\rangle$

$str(a_0, n) \wedge a = a_0 \wedge (0 \leq i < n) \wedge (0 \leq j < k-2) \wedge$
$\forall j \in 0..(k-2)\,(str(b,k) \subset str(a,i) \wedge (b[j] \neq b[j+1])) \wedge i \geq n$
$\Rightarrow$ $\langle$simplify, as i = n, and make j as i$\rangle$
$str(a_0, n) \wedge a = a_0 \wedge (0 \leq i < k-2) \wedge$
$\forall i \in 0..(k-2)\,(str(b,k) \subset str(a,n) \wedge (b[i] \neq b[i+1]))$
$\Leftrightarrow$ $\langle$simplify$\rangle$
$\forall i \in 0..(k-2)\,(str(b,k) \subset str(a,n) \wedge a = a_0 \wedge (b[i] \neq b[i+1]))$

2.5 Fifth implication:

$J \wedge 1 \leq i < n \Rightarrow I[^{i+1}/_i][^{k+1}/_k] \wedge (\exists h \in N\,(b[k][h] \neq a[i][h]) \vee \forall h \in N\,(a[i][h] = b[k][h]))$

$$
\begin{array}{ll}
 & J \wedge 1 \leq i < n \\
\Rightarrow & \quad \langle \text{definitions of } J \text{ and substitution} \rangle \\
 & I[^{i+1}/_i][^{k+1}/_k] \wedge i \geq 1 \wedge \\
 & (\exists h \in N\,(b[k][h] \neq a[i][h]) \vee \forall h \in N\,(a[i][h] = b[k][h])) \wedge 1 \leq i < n \\
\Rightarrow & \quad \langle \text{simplify,definitions of } I \text{ and substitution} \rangle \\
 & 0 < i+1 < n \wedge str(a_0, n) \wedge a = a_0 \wedge (0 \leq j < k-1) \\
 & \wedge \forall j \in 0..k-1\,(str(b, k+1) \subset str(a, i+1) \wedge b[j] \neq b[j+1]) \wedge 1 \leq i < n \\
 & \wedge (\exists h \in N\,(b[k][h] \neq a[i][h]) \vee \forall h \in N\,(a[i][h] = b[k][h])) \\
\Rightarrow & \quad \langle \text{simplify} \rangle \\
 & 1 < i+1 < n \wedge str(a_0, n) \wedge a = a_0 \wedge (0 \leq j < k-1) \\
 & \wedge \forall j \in 0..k-1\,(str(b, k+1) \subset str(a, i+1) \wedge b[j] \neq b[j+1]) \\
 & \wedge (\exists h \in N\,(b[k][h] \neq a[i][h]) \vee \forall h \in N\,(a[i][h] = b[k][h])) \\
\Leftrightarrow & \quad \langle \text{definitions of } I \text{ and substitution} \rangle \\
 & I[^{i+1}/_i][^{k+1}/_k] \wedge (\exists h \in N\,(b[k][h] \neq a[i][h]) \vee \forall h \in N\,(a[i][h] = b[k][h]))
\end{array}
$$

3 Task 3

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  #include "uniq.h"
5
6  unsigned int cmp(char *a, char *b) ;
7
8  unsigned int uniq(unsigned int n, char *a[], char *b[]) {
9      int k = 0;
10     /*Here opted for the more traditional for loop idiom instead of a while loop.*/
11     for(int i = 0; i < n; i ++) {
12         /*first if i = 0 branch to string copy the first element to string array b*/
13         if (i == 0) {
14             /*copy the first element into b[0]*/
15             b[i] = malloc(strlen(a[i]));
16             strcpy(b[i], a[i]);
17         } else {
18         /*first if else branch to string compare the first */
```

```
19              int h = 0;
20              /*In c implementation, checking individual characters
21              in both string arrays to preform the string compare.
22              Here opted for a and b pointer nullity checking, and
23               characters comparison as well as h increment */
24              for (; a != NULL && a[i][h] != '\0' && b != NULL && b[k][h] == a[i][h]; h++);
25              /*Check if exist h make a[i][h] and b[k][h] different */
26              if (a[i][h] != b[k][h]) {
27              /*increment k and alloction memory in c implementation*/
28                  b[++k] = malloc(strlen(a[i]));
29              /*assign a[i] to b[k]*/
30                  strcpy(b[k],a[i]);
31              }
32          }
33      }
34      return k;
35  }
```

In our C implementation, we opted for the more traditional **for** loop idiom instead of a **while** loop. It should be clear that our **for** loop captures the meaning of the entire WHILE loop in the toy language program. As for string copy(as proved and demonstrated in lecture, here has assuming the function of string copy is always true), we used a call of the C library function strcpy to implement the if branch (pseudo-code "$b[k] := a[i]$") that copies the content of array a into array b. (Cf. man strcpy.)