

# Assignment 1

z5141448

Ruofei HUANG

April 9, 2018

## Definations

### Two Dimensional Arrays Expression

To describe variants of two-dimensional arrays we write  $(b : k \mapsto j \mapsto x)$  instead of  $(b : k \mapsto (b[k] : j \mapsto x))$ . We use this new notation to state an instance of the array-assignment axiom we saw already

$$\{\phi^{(b:k \mapsto x)} / b\} b[k] := x \{\phi\}$$

for two-dimensional arrays:

$$\{\phi^{(b:k \mapsto j \mapsto x)} / b\} b[k][j] := x \{\phi\}$$

### String Length

A string  $S \in Letter^*$  which is an array of letters<sup>1</sup>. Also, string will be terminate by the null character which is a convention by the C programming language and we will follow this convention in this proof. We write  $|S|$  for the number of letters in the string. Formally, we define these two nothion inductively by

$$|S\ell| = |S| + \begin{cases} 1 & \text{if } \ell \neq ' \backslash 0' \\ 0 & \text{if } \ell = ' \backslash 0' \end{cases}$$

Also, by the convention of C we has this definition for  $S \in string$ .

$$S[|S|] = ' \backslash 0' \wedge \forall 0 \leq i < |S| (S[i] \neq ' \backslash 0')$$

---

<sup>1</sup>The letter here is a legal charater encode with ASCII, UTF-8 or other charater encoding standard.

### Precondition

In the toy language's programme, the string length must have a way to calculate. Here is the programme to calculate the String Length

$$a \in String$$

### Postcondition

$$a \in String \wedge aLength = |a|$$

### Programme

```
{a ∈ String}
{I0/aLength}
aLength := 0;
while a[aLength]! = ' \0' do
    {I ∧ a[aLength] ≠ ' \0'}
    {IaLength+1/aLength}
    aLength := aLength + 1;
od
{I ∧ a[aLength] = ' \0'}
{a ∈ String ∧ aLength = |a|}
```

Where

$$I = a \in String \wedge \forall k \in 0..(aLength - 1) (a[k] \neq ' \0')$$

**First Implication:**  $a \in String \Rightarrow I^{0/aLength}$

$$\begin{aligned} & a \in String \\ \Rightarrow & \quad \langle \text{The } \forall k \in 0..(0 - 1) (a[k] \neq ' \0') \text{ is always true.} \rangle \\ & a \in String \wedge \forall k \in 0..(0 - 1) (a[k] \neq ' \0') \\ \Leftrightarrow & \quad \langle \text{Definition of I.} \rangle \\ & I^{0/aLength} \end{aligned}$$

**Second Implication:**  $I \wedge a[aLength] \neq' \backslash 0' \Rightarrow I[aLength+1 / aLength]$

$$\begin{aligned}
& I \wedge a[aLength] \neq' \backslash 0' \\
\Leftrightarrow & \quad \langle \text{Definition of I.} \rangle \\
& a \in String \wedge \forall k \in 0..(aLength - 1) (a[k] \neq' \backslash 0') \wedge a[aLength] = ' \backslash 0' \\
\Leftrightarrow & \quad \langle \text{Merge same condition of } a[i] \neq' \backslash 0' \rangle \\
& a \in String \wedge \forall k \in 0..(aLength) (a[k] \neq' \backslash 0') \\
\Leftrightarrow & \quad \langle \text{Substitue back of I.} \rangle \\
& I[aLength+1 / aLength]
\end{aligned}$$

**Third Implication:**  $I \wedge a[aLength] = ' \backslash 0' \Rightarrow a \in String \wedge aLength = |a|$

$$\begin{aligned}
& I \wedge a[aLength] = ' \backslash 0' \\
\Leftrightarrow & \quad \langle \text{Definition of I.} \rangle \\
& a \in String \wedge \forall k \in 0..(aLength - 1) (a[k] \neq' \backslash 0') \wedge a[aLength] = ' \backslash 0' \\
\Leftrightarrow & \quad \langle \text{Definition of postcondition and the definition of String.} \rangle \\
& a \in String \wedge aLength = |a|
\end{aligned}$$

## String Equals

To describe two string  $a, b$  ( $a, b \in String$ ) are equals we write  $a = b$  when:

$$a = b \iff |a| = |b| \wedge \forall j \in 0..|a| (a[j] = b[j])$$

Similarly, we write:

$$a \neq b \iff \neg(a = b)$$

## Comparing String

After defining what is string equal, we need a pieces programme in toy language to do the dirty work which could compare two strings. Also we need a flag variable to store the truth value of  $a = b$

### Precondition

$$a, b \in String$$

### Postcondition

$$flag = (a = b)$$



## Programme

```
 $\{a, b \in \text{String}\}$   
 $\{I^{[\text{TRUE}/\text{flag}]}[{}^0/i]\}$   
 $i := 0;$   
 $\text{flag} := 1;$   
 $\{I\}$   
if  $|a| = |b|$  then  
   $\{I \wedge |a| = |b|\}$   
   $\{J\}$   
  while  $i \leq |a|$  do  
     $\{J \wedge i \leq |a|\}$   
     $\{K\}$   
    if  $a[i] = b[i]$  then  
       $\{K \wedge a[i] = b[i]\}$   
      skip  
    else  
       $\{K \wedge a[i] \neq b[i]\}$   
       $\{J^{[\text{FALSE}/\text{flag}]}[{}^{i+1}/i]\}$   
       $\text{flag} := 0;$   
       $\{J^{[i+1]/i}\}$   
    fi  
     $\{J^{[i+1]/i}\}$   
     $i := i + 1;$   
     $\{J\}$   
  od  
   $\{J \wedge i > |a|\}$   
   $\{I\}$   
else  
   $\{I \wedge |a| \neq |b|\}$   
   $\{I^{[\text{FALSE}/\text{flag}]}\}$   
   $\text{flag} := 0;$   
   $\{I\}$   
fi  
 $\{I\}$   
 $\{\text{flag} = (a = b)\}$ 
```

Where TRUE= 1 , FALSE= 0, and

$$I = a, b \in String \wedge flag = (|a| = |b| \wedge 0 \leq i \leq (|a| + 1) \wedge \forall k \in 0..(i - 1) (a[i] = b[i]))$$

$$J = a, b \in String \wedge flag = (0 \leq i \leq (|a| + 1) \wedge \forall k \in 0..(i - 1) (a[i] = b[i]))$$

$$K = \left( \begin{array}{l} a[i] \neq b[i] \Rightarrow I[\text{FALSE}/flag][^{i+1}/i] \\ a[i] = b[i] \Rightarrow I[^{i+1}/i] \end{array} \right)$$

**First Implication:**  $a, b \in String \Rightarrow I[\text{true}/flag][^0/i]$

$$\begin{aligned} & a, b \in String \\ \Rightarrow & \langle \text{The append junction is always true.} \rangle \\ & a, b \in String \wedge |a| = |b| \wedge 0 \leq i \leq 0 \wedge \forall k \in 0..(i - 1) (a[i] = b[i]) \\ \Rightarrow & \langle flag = \text{TRUE and TRUE} = \text{TRUE} \rangle \\ & a, b \in String \wedge flag = (|a| = |b| \wedge 0 \leq i \leq 0 \wedge \forall k \in 0..(i - 1) (a[i] = b[i])) \\ \Leftrightarrow & \langle \text{Defination of I.} \rangle \\ & I[\text{TRUE}/flag] \end{aligned}$$

**Second Implication:**  $I \wedge |a| = |b| \Rightarrow J \wedge [^0/i]$

$$\begin{aligned} & I \wedge |a| = |b| \\ \Rightarrow & \langle \text{Definiation of I. } i = 0 \text{ when right junction is always true.} \rangle \\ & a, b \in String \wedge flag = (|a| = |b| \wedge 0 \leq 0 \leq (|a| + 1) \\ & \quad \wedge \forall k \in 0..(i - 1) (a[i] = b[i])) \\ \Rightarrow & \langle \text{Definition of J.} \rangle \\ & J \wedge [^0/i] \end{aligned}$$

**Third Implication:**  $J \wedge i \leq |a| \Rightarrow K$

$$\begin{aligned}
& J \wedge i \leq |a| \\
\Leftrightarrow & \quad \langle \text{Definition of } J \rangle \\
& a, b \in \text{String} \wedge \text{flag} = (0 \leq i \leq (|a| + 1) \wedge \forall k \in 0..(i - 1) (a[i] = b[i])) \wedge i \leq |a| \\
\Rightarrow & \quad \langle \text{By definition of two string is equal, two case to consider.} \rangle \\
& \left( \begin{array}{l} a[i] \neq b[i] \Rightarrow a, b \in \text{String} \wedge \text{flag} = \text{FALSE} \\ a[i] = b[i] \Rightarrow a, b \in \text{String} \wedge \text{flag} = (|a| = |b| \\ \wedge 0 \leq i + 1 \leq (|a| + 2) \wedge \forall k \in 0..i (a[i + 1] = b[i + 1])) \wedge i \leq |a| \end{array} \right) \\
\Rightarrow & \quad \langle \text{By definition of two string is equal, two case to consider.} \rangle \\
& \left( \begin{array}{l} a[i] \neq b[i] \Rightarrow a, b \in \text{String} \wedge \text{flag} = \text{FALSE} \\ a[i] = b[i] \Rightarrow a, b \in \text{String} \wedge \text{flag} = (|a| = |b| \\ \wedge 0 \leq i + 1 \leq (|a| + 1) \wedge \forall k \in 0..i (a[i + 1] = b[i + 1])) \end{array} \right) \\
\Leftrightarrow & \quad \langle \text{Definition of } I \rangle \\
& \left( \begin{array}{l} a[i] \neq b[i] \Rightarrow I[\text{FALSE}/\text{flag}][i+1/i] \\ a[i] = b[i] \Rightarrow I[i+1/i] \end{array} \right) \\
\Leftrightarrow & \quad \langle \text{Defination of } K \rangle \\
& K
\end{aligned}$$

**Forth Implication:**  $I \wedge |a| \neq |b| \Rightarrow I[\text{false}/\text{flag}]$

$$\begin{aligned}
& I \wedge |a| \neq |b| \\
\Rightarrow & \quad \langle \text{Definition of two string is not equal.} \rangle \\
& I \wedge |a| \neq |b| \wedge a \neq b \\
\Rightarrow & \quad \langle \text{flag} = \text{FALSE and FALSE} = \text{FALSE} \rangle \\
& a, b \in \text{String} \wedge \text{flag} = (a = b) \wedge a \neq b \\
\Leftrightarrow & \quad \langle \text{Definition of I.} \rangle \\
& I[\text{FALSE}/\text{flag}]
\end{aligned}$$

**Fifth Implication:**  $I \Rightarrow \text{flag} = (a = b)$

$$\begin{aligned}
& I \\
\Leftrightarrow & \quad \langle \text{Defination of the purpose of flag} \rangle \\
& \text{flag} = (a = b)
\end{aligned}$$

## String Assign

To assign a string to another string array, we will denote as

$$a := b$$

instead of a long programme of our toy language:

```

{a, b ∈ String}
{I0/i}
i := 0;
{I}
while i ≤ |b| do
  {I ∧ i ≤ |b|}
  {Ii+1/i}[a:i→b[i]/a]
  a[i] := b[i];
  {Ii+1/i}
  i := i + 1;
  {I}
od;
{I ∧ i > |b|}
{a, b ∈ String ∧ a = b}

```

when our invariant is

$$I = a, b \in \text{String} \wedge 0 \leq i \leq (|b| + 1) \wedge \forall k \in 0..(i - 1) (a[k] = b[k])$$

Here are the proofs of the implications:

**First Implication for String assign:**  $a, b \in \text{String} \Rightarrow I^{[0/i]}$

$$\begin{aligned}
& a, b \in \text{String} \\
\Rightarrow & \quad \langle \text{using } |b| \in \mathbb{N} \text{ and realising that the last conjunct is vacuously true} \rangle \\
& a, b \in \text{String} \wedge 0 \leq 0 \leq (|b| + 1) \wedge \forall k \in 0..(0 - 1) (a[k] = b[k]) \\
\Leftrightarrow & \quad \langle \text{definition of I and substitution} \rangle \\
& I^{[0/i]}
\end{aligned}$$

**Second Implication**  $I \wedge i \leq |b| \Rightarrow I^{[i+1/i]}[<sup>a:i→b[i]</sup>/a]$

We first look at the LHS:

$$\begin{aligned}
& I \wedge i \leq |b| \\
\Leftrightarrow & \quad \langle \text{Substitute I} \rangle \\
& a, b \in \text{String} \wedge 0 \leq i \leq (|b| + 1) \wedge \forall k \in 0..(i - 1) (a[k] = b[k]) \wedge i \leq |b| \\
\Leftrightarrow & \quad \langle \text{Conjunct } i \leq (|b| + 1) \text{ and } i \leq |b| \rangle \\
& a, b \in \text{String} \wedge 0 \leq i \leq |b| \wedge \forall k \in 0..(i - 1) (a[k] = b[k])
\end{aligned}$$



We then expand RHS:

$$\begin{aligned}
& I^{[i+1]/i}[a:i \rightarrow b[i]/a] \\
\Leftrightarrow & \quad \langle \text{substitute } i = i + 1 \text{ and } a[i] = b[i] \text{ by definition} \rangle \\
& a, b \in \text{String} \wedge 0 \leq i + 1 \leq (|b| + 1) \wedge \forall k \in 0..((i + 1) - 1) (a[k] = b[k]) \wedge a[i] := b[i]
\end{aligned}$$

We then have a clear imply

$$\begin{aligned}
& a, b \in \text{String} \wedge 0 \leq i \leq |b| \wedge \forall k \in 0..(i - 1) (a[k] = b[k]) \\
\Rightarrow & \quad \langle i \leq |b| \Rightarrow i + 1 \leq |b| + 1 \text{ and } a[i] := b[i] \rangle \\
& a, b \in \text{String} \wedge 0 \leq i + 1 \leq (|b| + 1) \wedge \forall k \in 0..((i + 1) - 1) (a[k] = b[k]) \wedge a[i] := b[i]
\end{aligned}$$

**Third Implication**  $I \wedge i > |b| \Rightarrow a, b \in \text{String} \wedge a = b$

$$\begin{aligned}
& I \wedge i > |b| \\
\Leftrightarrow & \quad \langle \text{substitution of I} \rangle \\
& a, b \in \text{String} \wedge 0 \leq i \leq (|b| + 1) \wedge \forall k \in 0..(i - 1) (a[k] := b[k]) \wedge i > |b| \\
\Leftrightarrow & \quad \langle i > |b| \text{ and } i \leq (|b| + 1) \text{ with some calculation} \rangle \\
& a, b \in \text{String} \wedge \forall k \in 0..|b| (a[k] := b[k]) \\
\Rightarrow & \quad \langle \text{Definition of two string equal} \rangle \\
& a, b \in \text{String} \wedge a = b
\end{aligned}$$

## 1 Task 1

Since we have define some manipulation of String, we can see a string as a whole. So the input is an array of String. Also , the ouput is store in  $b$  which is an empty array (type is  $\text{String}^*$  too ). Hence we can define our precondition as:

$$a, b \in \text{String}^* \wedge |a| = n$$

As the post condition as:

$$a, b \in \text{String} \wedge \forall i < n (a[i] = b[m(a, i)])$$

Where  $m$  is a mapping function define recursively as follow:

$$m(a, i) = \begin{cases} 0 & \text{if } i = 0 \\ m(a, i - 1) & \text{if } a[i] = a[i - 1] \\ m(a, i - 1) + 1 & \text{if } a[i] \neq a[i - 1] \end{cases}$$



## 2 Task 2

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

$$\begin{aligned}
& \{a, b \in \text{String}^* \wedge |a| = n\} & (1) \\
& \{J\} & (2) \\
& \text{if } |a| > 0 \text{ then} & (3) \\
& \quad \{J \wedge |a| > 0\} & (4) \\
& \quad \{I[1/j][1/i][b:0 \mapsto a[0]/b]\} & (5) \\
& \quad b[0] := a[0]; & (6) \\
& \quad \{I[1/j][1/i]\} & (7) \\
& \quad i = 1; j = 1; & (8) \\
& \quad \{I\} & (9) \\
& \text{else} & (10) \\
& \quad \{J \wedge |a| \leq 0\} & (11) \\
& \quad \{I[0/i][0/j]\} & (12) \\
& \quad i := 0; j := 0; & (13) \\
& \quad \{I\} & (14) \\
& \text{fi} & (15) \\
& \{I\} & (16) \\
& \text{while } i < |a| \text{ do} & (17) \\
& \quad \{I \wedge i < |a|\} & (18) \\
& \quad \{K\} & (19) \\
& \quad \text{if } a[i] \neq a[i-1] \text{ then} & (20) \\
& \quad \quad \{K \wedge a[i] \neq a[i-1]\} & (21) \\
& \quad \quad \{I[i+1/i][j+1/j][b:j \mapsto a[i]/b]\} & (22) \\
& \quad \quad b[j] := a[i]; & (23) \\
& \quad \quad \{I[i+1/i][j+1/j]\} & (24) \\
& \quad \quad j := j + 1; & (25) \\
& \quad \quad \{I[i+1/i]\} & (26) \\
& \quad \text{else} & (27) \\
& \quad \quad \text{skip} & (28) \\
& \quad \text{fi} & (29) \\
& \quad \{I[i+1/i]\} & (30) \\
& \quad i := i + 1 & (31) \\
& \quad \{I\} & (32) \\
& \text{od} & (33) \\
& \{I \wedge i \geq |a|\} & (34) \\
& \{a, b \in \text{String} \wedge \forall i < n (a[i] = b[m(a, i)])\} & (35)
\end{aligned}$$

Here are the invariants of this programme <sup>2</sup>:

$$\begin{aligned}
I &= a, b \in \text{String}^* \wedge |a| = n \wedge 0 \leq i \leq |a| \wedge \forall k \in 0..(i-1) (a[k] = b[m(a, k)]) \\
J &= \left( \begin{array}{l} |a| > 0 \Rightarrow I[1/j][1/i][b:0 \rightarrow a[0]/b] \\ |a| \leq 0 \Rightarrow I[0/j][0/i] \end{array} \right) \\
K &= \left( \begin{array}{l} a[i] \neq a[i-1] \Rightarrow I[i+1/i][j+1/j][b:j \rightarrow a[i]/b] \\ a[i] = a[i-1] \Rightarrow I[i+1/i] \end{array} \right)
\end{aligned}$$

## 2.1 First Implication: $a, b \in \text{String}^* \wedge |a| = n \Rightarrow J$

$$\begin{aligned}
&a, b \in \text{String}^* \wedge |a| = n \\
\Rightarrow &\quad \langle n \in \mathbb{N}, \text{substitution of } i, j \rangle \\
&\left( \begin{array}{l} |a| > 0 \Rightarrow a, b \in \text{String}^* \wedge |a| = n \wedge 0 \leq 1 \leq |a| \wedge \forall k \in 0..(1-1) (a[k] = b[m(a, k)]) \\ |a| = 0 \Rightarrow a, b \in \text{String}^* \wedge |a| = n \wedge 0 \leq 0 \leq |a| \wedge \forall k \in 0..(0-1) (a[k] = b[m(a, k)]) \end{array} \right) \\
\Rightarrow &\quad \langle \text{Substitution of I} \rangle \\
&\left( \begin{array}{l} |a| > 0 \Rightarrow I[1/j][1/i][b:0 \rightarrow a[0]/b] \\ |a| \leq 0 \Rightarrow I[0/j][0/i] \end{array} \right) \\
\Leftrightarrow &\quad \langle \text{Definition of J} \rangle \\
&J
\end{aligned}$$

---

<sup>2</sup>The invariant is following the case study in week 8, might not be true for the stuff we study for now.  
But I have to use this tool otherwise I couldn't continue this proof

## 2.2 Second Implication: $I \wedge i < |a| \Rightarrow K$

$$\begin{aligned}
& I \wedge i < |a| \\
\Leftrightarrow & \langle \text{Definition of } I \rangle \\
& a, b \in \text{String}^* \wedge |a| = n \wedge 0 \leq i \leq |a| \wedge \forall k \in 0..(i-1) (a[k] = b[m(a, k)]) \wedge i < |a| \\
\Rightarrow & \langle m(a, i) \text{ need to consider two situation of } a[i] \text{ (} a[i] = a[i-1] \text{ and } a[i] \neq a[i-1]) \rangle \\
& \left( \begin{array}{l} a[i] \neq a[i-1] \Rightarrow a, b \in \text{String}^* \wedge |a| = n \wedge \\ 0 \leq i+1 \leq |a| \wedge \forall k \in 0..(i-1) (a[k] = b[m(a, k)]) \\ \wedge a[i] = b[j] \\ a[i] = a[i-1] \Rightarrow a, b \in \text{String}^* \wedge |a| = n \wedge \\ 0 \leq i+1 \leq |a| \wedge \forall k \in 0..(i-1) (a[k] = b[m(a, k)]) \\ \wedge a[i] = b[j-1] \end{array} \right) \\
\Leftrightarrow & \langle \text{By the recursively definition of } m(a, i) \rangle \\
& \left( \begin{array}{l} a[i] \neq a[i-1] \Rightarrow a, b \in \text{String}^* \wedge |a| = n \wedge \\ 0 \leq i+1 \leq |a| \wedge \forall k \in 0..(i-1) (a[k] = b[m(a, k)]) \\ \wedge a[i] = b[m(a, i)] \\ a[i] = a[i-1] \Rightarrow a, b \in \text{String}^* \wedge |a| = n \wedge \\ 0 \leq i+1 \leq |a| \wedge \forall k \in 0..(i-1) (a[k] = b[m(a, k)]) \\ \wedge a[i] = b[m(a, i)] \end{array} \right) \\
\Leftrightarrow & \langle \text{Merge two cases of } k \text{ (} k = i \text{ and } k \in 0..(i-1) \text{) .} \rangle \\
& \left( \begin{array}{l} a[i] \neq a[i-1] \Rightarrow a, b \in \text{String}^* \wedge |a| = n \wedge \\ 0 \leq i+1 \leq |a| \wedge \forall k \in 0..(i) (a[k] = b[m(a, k)]) \\ a[i] = a[i-1] \Rightarrow a, b \in \text{String}^* \wedge |a| = n \wedge \\ 0 \leq i+1 \leq |a| \wedge \forall k \in 0..(i) (a[k] = b[m(a, k)]) \end{array} \right) \\
\Leftrightarrow & \langle \text{Substitue back by definition of I.} \rangle \\
& \left( \begin{array}{l} a[i] \neq a[i-1] \Rightarrow I^{[i+1/i]}[j+1/j][b:j \rightarrow a[i]/b] \\ a[i] = a[i-1] \Rightarrow I^{[i+1/i]} \end{array} \right) \\
\Leftrightarrow & \langle \text{Definition of K} \rangle \\
& K
\end{aligned}$$

The meaning of "By the recursively definition of  $m(a, i)$ " is when  $a[i] = a[i-1]$ ,  $a[i] = b[m(a, i)] = b[m(a, i-1)]$ , which  $m(a, i-1) = j-1$  since  $j = |b|$ . In another case,  $a[i] \neq a[i-1]$ ,  $b[j] = a[i]$  (where  $|b| = j+1$ )

### 2.3 Third Implication:

$$I \wedge i \geq |a| \Rightarrow a, b \in \text{String} \wedge \forall i < n (a[i] = b[m(a, i)])$$

$$\begin{aligned} & I \wedge i \geq |a| \\ \Leftrightarrow & \langle \text{Definitions of I} \rangle \\ & a, b \in \text{String}^* \wedge |a| = n \wedge 0 \leq i \leq |a| \wedge \forall k \in 0..(i-1) (a[k] = b[m(a, k)]) \wedge i \geq |a| \\ \Leftrightarrow & \langle i \leq |a| \wedge i \geq |a| \Leftrightarrow i = |a| \rangle \\ & a, b \in \text{String}^* \wedge |a| = n \wedge \forall k \in 0..(|a| - 1) (a[k] = b[m(a, k)]) \\ \Rightarrow & \langle |a| = n \text{ and } i \in \mathbb{N} \rangle \\ & a, b \in \text{String}^* \wedge \forall i \in 0..(n-1) (a[i] = b[m(a, i)]) \\ \Leftrightarrow & \langle \text{Defination of post condition.} \rangle \\ & a, b \in \text{String} \wedge \forall i < n (a[i] = b[m(a, i)]) \end{aligned}$$

## 3 Task 3

```
1  int length_str(char *a){
2      int i = 0;
3      while(a[i]!='\0'){
4          i = i+1;
5      }
6      return i;
7  }
8
9  void copy_str(char *a, char *b){
10     // copy a particular string from a to b
11     int i = 0;
12     while(i <= length_str(a) ){
13         b[i] = a[i];
14         i++;
15     }
16 }
17
18 int compare_str(char *a, char *b){
19     int flag = 1;
20     if(length_str(a) == length_str(b))
21     {
22         int i = 0;
23         while (i <= length_str(a)){
24             if (a[i] != b[i]){
```

```

25         flag = 0;
26     }
27     else{
28         // skip
29     }
30     i++;
31 }
32 }
33 else{
34     flag =0;
35 }
36 return flag;
37 }
38
39 unsigned int uniq(unsigned int n, char *a[], char *b[]){
40     int i,j;
41     if(n > 0){
42         copy_str(a[0],b[0]);
43         i =j =1;
44     }
45     else{
46         i = j=0;
47     }
48     while(i<n){
49         if (!compare_str(a[i], a[i-1])){
50             copy_str(a[i],b[j]);
51             j++;
52         }
53         else{
54             // skip
55         }
56         i++;
57     }
58     return j;
59 }

```

## 4 Task 4