BURROWS WHEELER ALGORITHM

BWT is simple way to sort and group data together which makes compressing easier. The similar letters are grouped together. It is also known as block sorting compression.

First step – It is used to create a table by rotating the last letter of the text.

Second step – sort the table with sorting algorithm in alphabetical order.

Third step- Take the last column of the sorted text that is the output of BWT.

BWT with example

Let us take the word at company$ with $ as special character in End of the Text

Step 1: Rotating last letter

Company$

Ompany$c

Mpany$co

Pany$com

Any$comp

Ny$compa

Y$compan

$company

Step 2: arrange in alphabetical order, special characters have first priority.

$company

any$comp

company$

mpany$co

ny$compa

ompany$c

pany$com

y$compan

Step 3- the last coloumn is BWT

Yp$oacmn

C++ implementation of BWT algorithm

We use simple rotation and sorting functions in the following c++ implementation

The text to be given as input

“Curneu MedTech Innovation is a health care technology firm based at Heidelberg, Germany. We work on a motive of building affordable and

innovative healthcare solutions that address the clinical needs thereby bringing

better lives for the needy.”

Rotation of the string

**void** rotate**(**std::**string** &a**)** **{**  
 **char** t = a**[**a.**length()** - **1]**;  
 **for** **(int** i = a.**length()** - **1**; i > **0**; i--**)** **{**  
 a**[**i**]** = a**[**i - **1]**;  
 **}**  
 a**[0]** = t;  
**}**

the above programs rotates through the text

The STX and ETX are added with special characters $ and @  because it marks the point where the character cycle starts and ends

std::string makePrintable(const std::string &s) {

auto ls = s;

for (auto &c : ls) {

if (c == STX) {

c = '@';

} else if (c == ETX) {

c = '$';

}

}

return ls;

}

Sorting the string

std::**sort(**table.**begin()**, table.**end())**;  
   
 std::**string** out;  
 **for** **(auto** &s : table**)** **{**  
 out += s**[**s.**length()** - **1]**;  
 **}**  
 **return** out;  
**}**

Extract the Last element

std::string ss;

ss += STX;

ss += s;

ss += ETX;

std::vector<std::string> table;

for (size\_t i = 0; i < ss.length(); i++) {

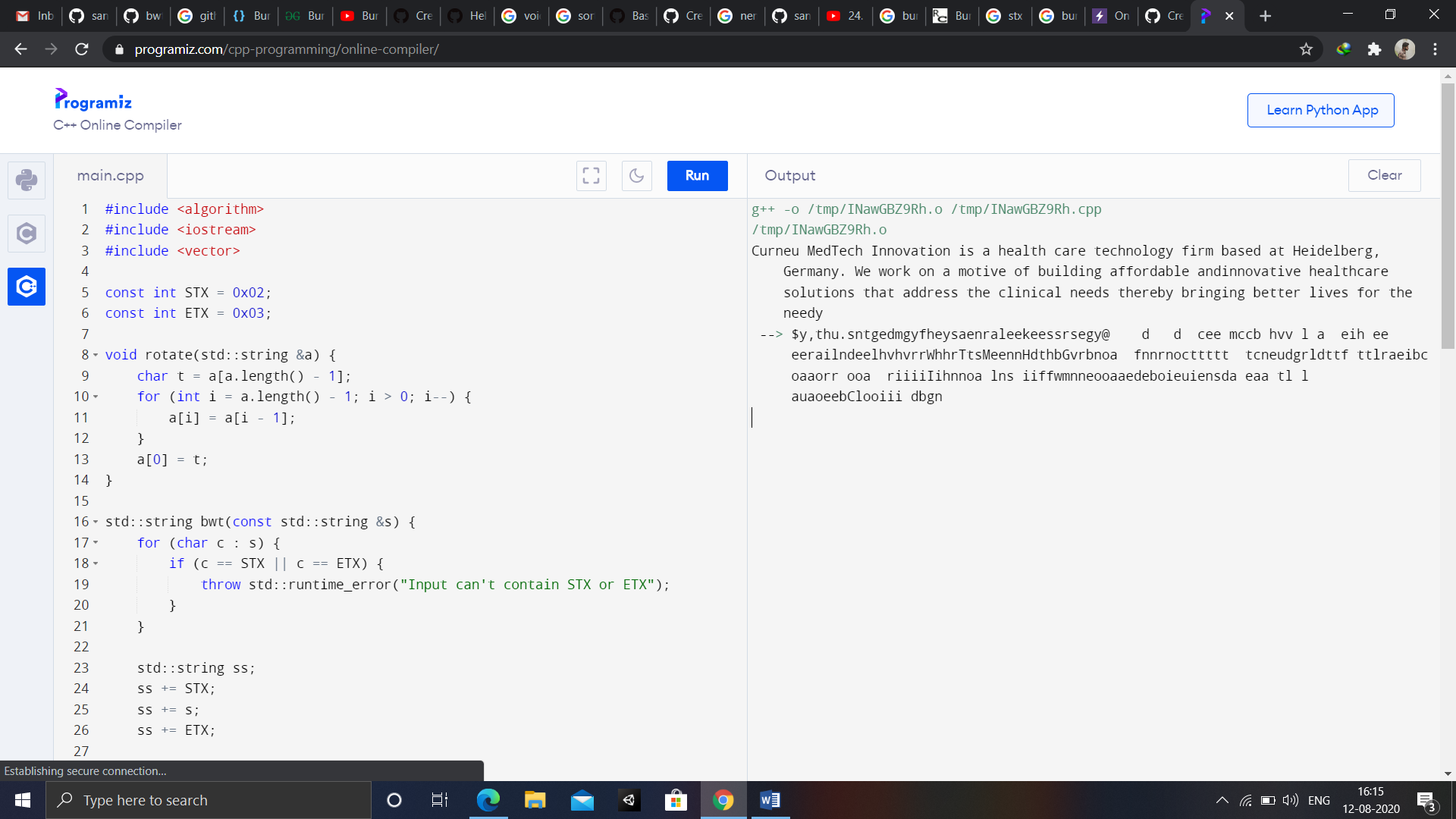
table.push\_back(ss);

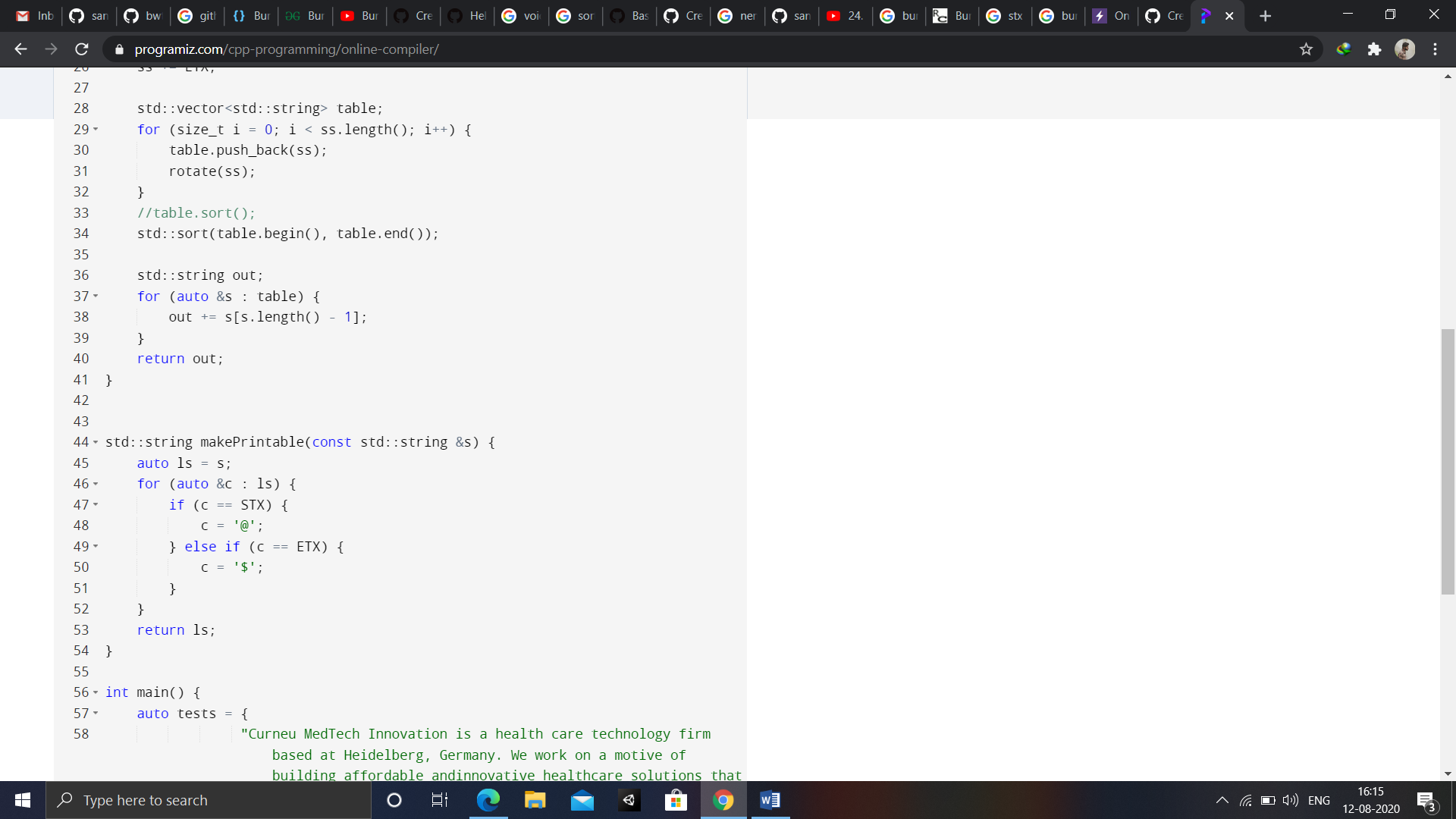
rotate(ss);

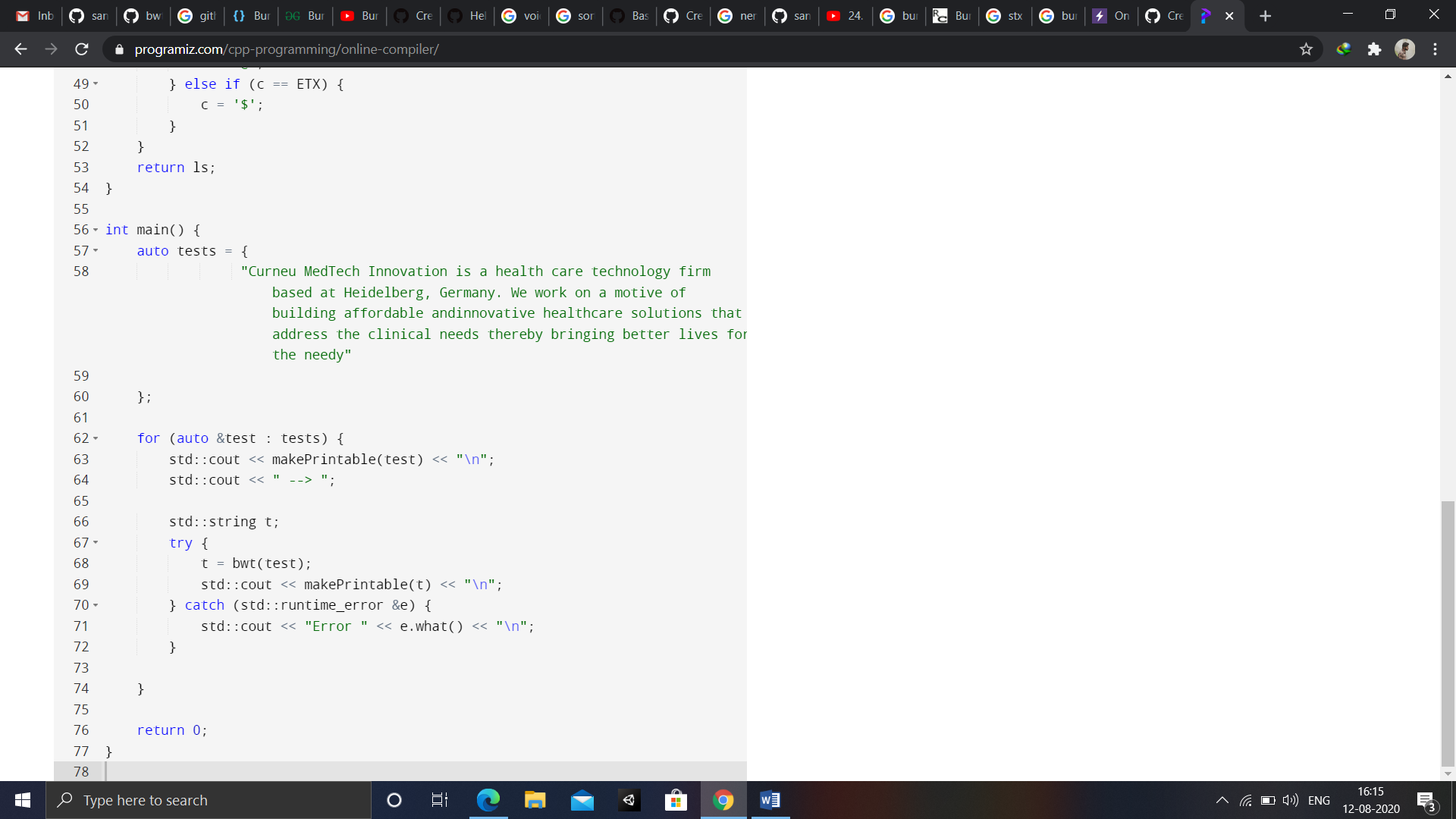
}

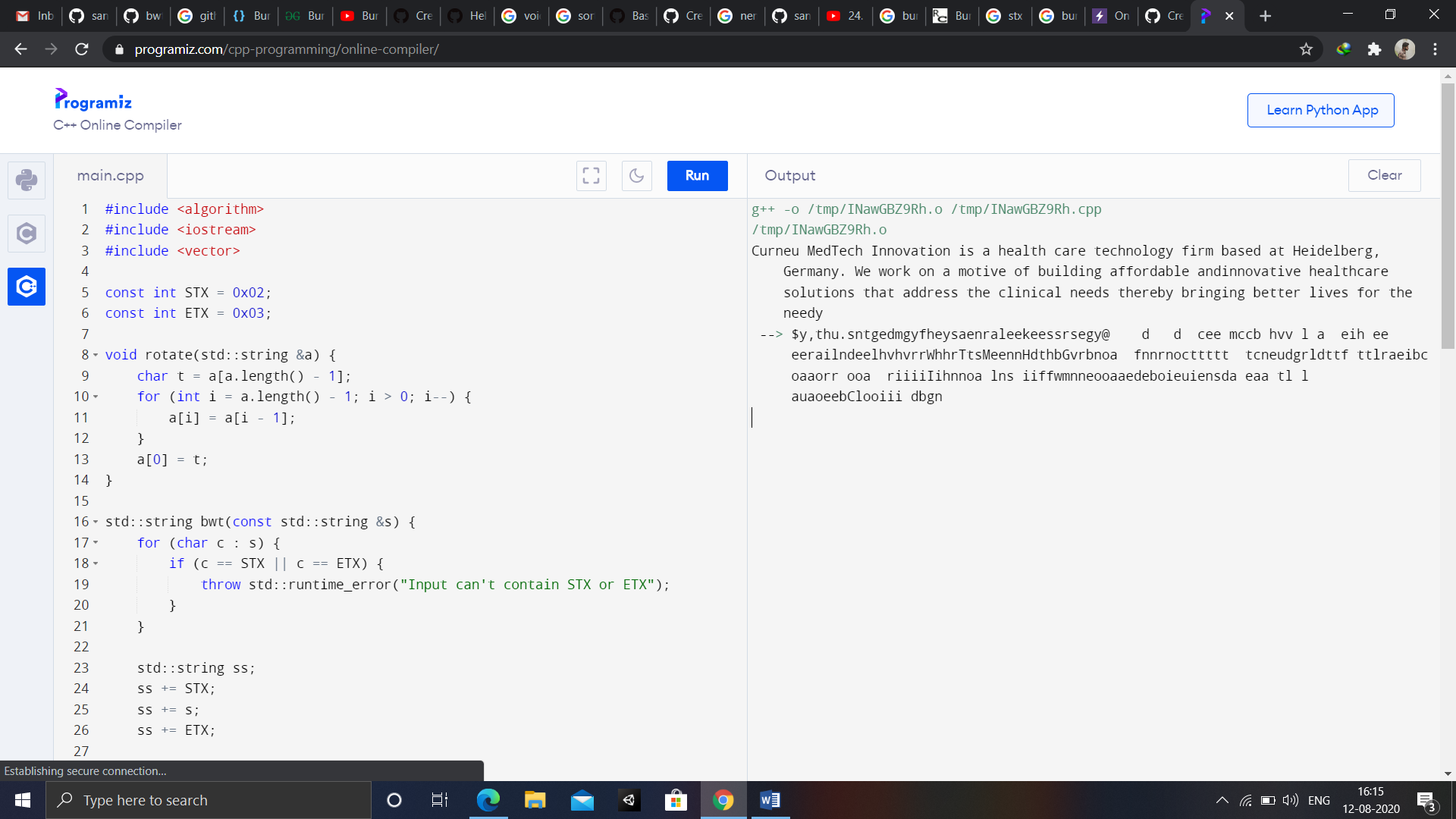
The above code extracts the last element from the columns

CODING AND OUTPUT:









OUTPUT:

$y,thu.sntgedmgyfheysaenraleekeessrsegy@ d d cee mccb hvv l a eih ee eerailndeelhvhvrrWhhrTtsMeennHdthbGvrbnoa fnnrnocttttt tcneudgrldttf ttlraeibc oaaorr ooa riiiiIihnnoa lns iiffwmnneooaaedeboieuiensda eaa tl l auaoeebClooiii dbgn