Yifan Yang

40038814

assignment 3

1.

MyArray=["Adam","Eve","Mark","Franklin","John"]

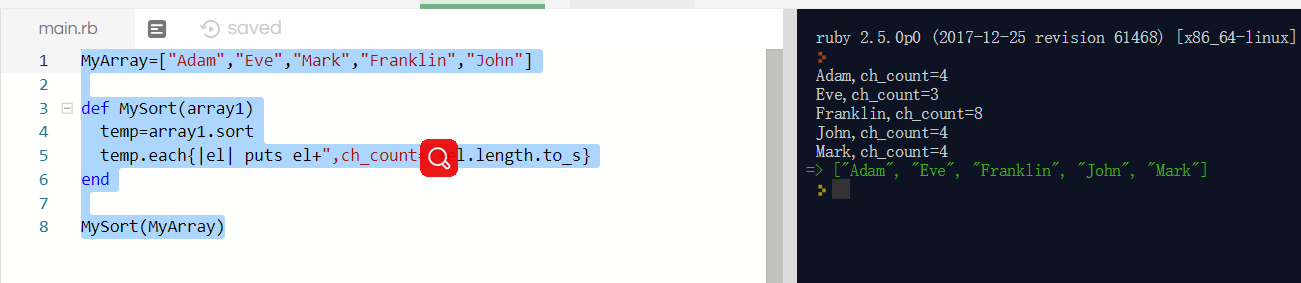
def MySort(array1)

temp=array1.sort

temp.each{|el| puts el+",ch\_count="+el.length.to\_s}

end

MySort(MyArray)



2.

def calcARI(myTXT)

totalChar=0

totalSpace=1

totalDot=0

temp=File.open(myTXT)

for line in temp

s=line.delete(' ')

totalChar+=s.length

totalSpace+=line.count(' ')

totalDot+=line.count('.')

end

puts "Total # of characters: "+totalChar.to\_s

puts "Total # of words: "+totalSpace.to\_s

puts "Total # of sentences: "+totalDot.to\_s

ari=4.71\*(totalChar/totalSpace)+0.5\*(totalSpace/totalDot)-21.43

puts "Automated Readability Index: "+ari.to\_s

print "Grade level: "

case ari

when 1...2

print "5-6(kindergarten)"

when 2...3

print "6-7(first/second grade)"

when 3...4

print "7-9(third grade)"

when 4...5

print "9-10(fourth grade)"

when 5...6

print "10-11(fifth grade)"

when 6...7

print "11-12(sixth grade)"

when 7...8

print "12-13(seventh grade)"

when 8...9

print "13-14(eighth grade)"

when 9...10

print "14-15(ninth grade)"

when 10...11

print "15-16(tenth grade)"

when 11...12

print "16-17(eleventh grade)"

when 12...13

print "17-18(twelfth grade)"

when 13...14

print "18-24(college student)"

else

print "24+(professor)"

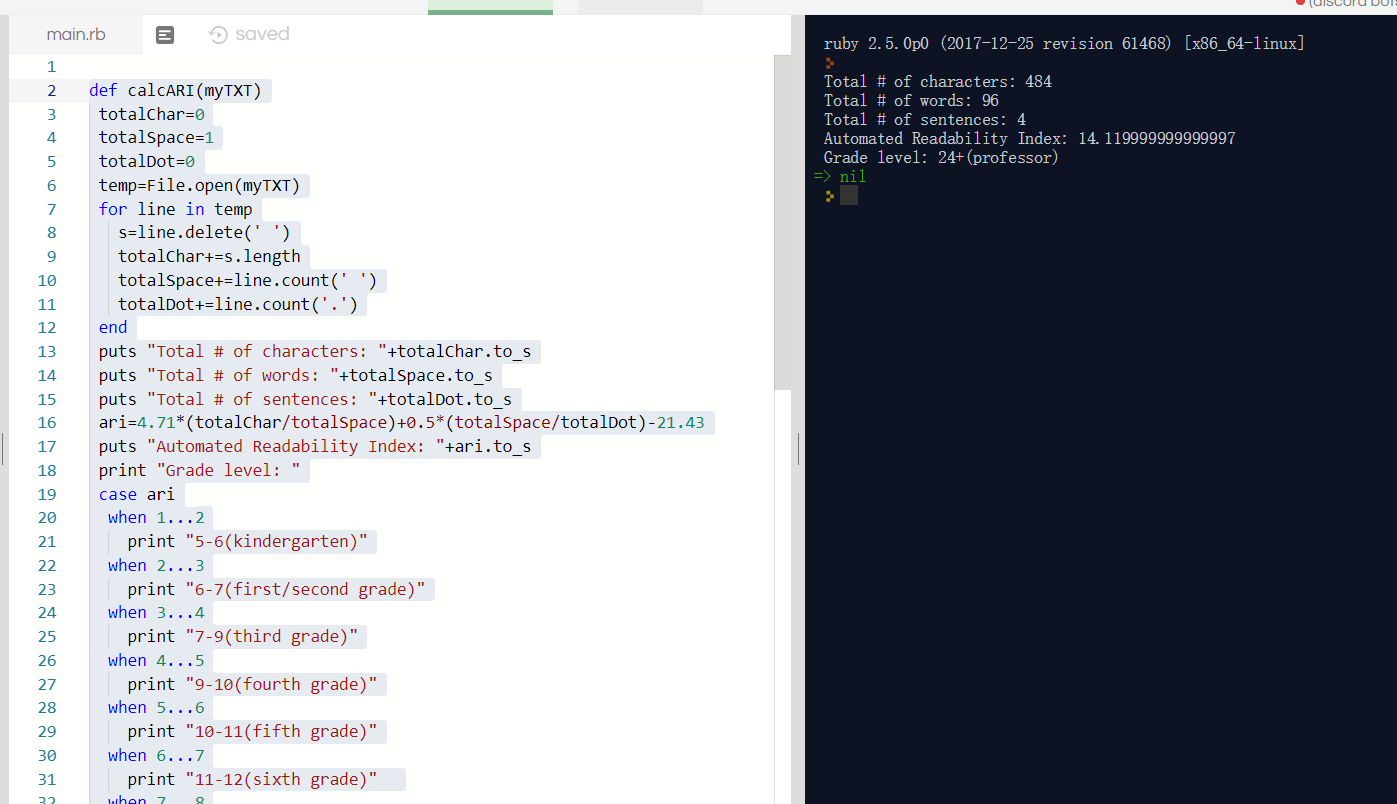
end

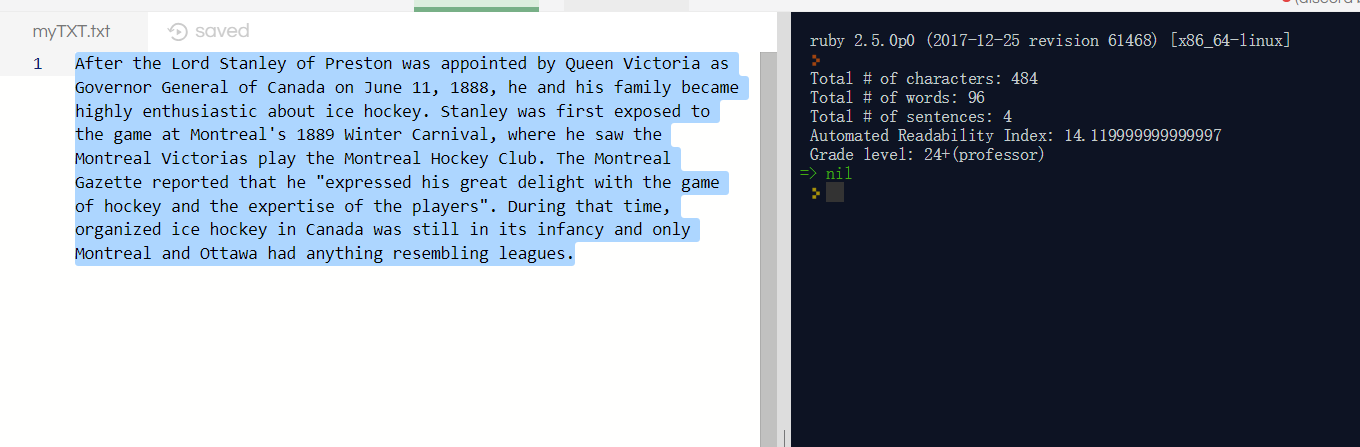
end

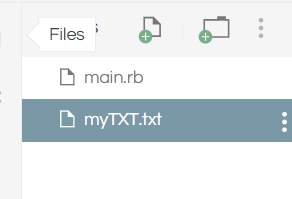
calcARI("myTXT.txt")

txt part：

After the Lord Stanley of Preston was appointed by Queen Victoria as Governor General of Canada on June 11, 1888, he and his family became highly enthusiastic about ice hockey. Stanley was first exposed to the game at Montreal's 1889 Winter Carnival, where he saw the Montreal Victorias play the Montreal Hockey Club. The Montreal Gazette reported that he "expressed his great delight with the game of hockey and the expertise of the players". During that time, organized ice hockey in Canada was still in its infancy and only Montreal and Ottawa had anything resembling leagues.







3.

def getData(myTXT)

inputData=[]

temp=File.open(myTXT)

for line in temp

inputData<< ","+line.chomp+","

end

return inputData

end

class BasicFrame

@@total = 0

def initialize(inputData)

@@total += 1

end

def to\_s

return ""

end

def BasicFrame.total

return "Number of cars: #@@total"

end

end

class Km<BasicFrame

attr\_accessor :km

def initialize (inputData)

super(inputData)

a=/\d+km/i.match(inputData)

@km=a[0]

end

def to\_s

return super+@km

end

end

class Car\_maker<Km

attr\_accessor :car\_maker

def initialize (inputData)

super(inputData)

a=/,Honda,|,Toyota,|,Mercedes,|,BMW,|,Lexus,/i.match(inputData)

@car\_maker=a[0].delete(",")

end

def to\_s

return super+@car\_maker

end

end

class Type<Car\_maker

attr\_accessor :type

def initialize (inputData)

super(inputData)

a=/,Sedan,|,coupe,|,hatchback,|,station,|,SUV,/i.match(inputData)

@type=a[0].delete(",")

end

def to\_s

return super+@type

end

end

class Transimission<Type

attr\_accessor :transimission

def initialize (inputData)

super(inputData)

a=/,auto,|,manual,|,steptronic,/i.match(inputData)

@transimission=a[0].delete(",")

end

def to\_s

return super+@transimission

end

end

class Drivetrain<Transimission

attr\_accessor :drivetrain

def initialize (inputData)

super(inputData)

a=/,fwd,|,rwd,|,awd,/i.match(inputData)

@drivetrain=a[0].delete(",")

end

def to\_s

return super+@drivetrain

end

end

class Status<Drivetrain

attr\_accessor :status

def initialize (inputData)

super(inputData)

a=/,used,|,new,/i.match(inputData)

@status=a[0].delete(",")

end

def to\_s

return super+@status

end

end

class Year<Status

attr\_accessor :year

def initialize (inputData)

super(inputData)

a=/,\d{4},/i.match(inputData)

@year=a[0].delete(",")

end

def to\_s

return super+@year

end

end

class Trim<Year

attr\_accessor :trim

def initialize (inputData)

super(inputData)

a=/,[A-Z]{2},/i.match(inputData)

@trim=a[0].delete(",")

end

def to\_s

return super+@trim

end

end

class Sets<Trim

attr\_accessor :sets

def initialize (inputData)

super(inputData)

a=/{.\*}/i.match(inputData)

@sets=a[0]

end

def to\_s

return super+@sets

end

end

class Fuel<Sets

attr\_accessor :fuel

def initialize (inputData)

super(inputData)

a=/,(\d\*|\d\*\.\d)L\/\d\*km,/i.match(inputData)

@fuel=a[0].delete(",")

end

def to\_s

return super+@fuel

end

end

class Model<Fuel

attr\_accessor :model

def initialize (inputData)

super(inputData)

tempa=inputData.gsub!(@km, "")

tempa=inputData.gsub!(@car\_maker, "")

tempa=inputData.gsub!(@drivetrain, "")

tempa=inputData.gsub!(@trim, "")

tempa=inputData.gsub!(@sets, "")

tempa=inputData.gsub!(@status, "")

tempa=inputData.gsub!(@fuel, "")

tempa=inputData.gsub!(@type, "")

tempa=inputData.gsub!(@transimission, "")

tempa=inputData.gsub!(@year, "")

a=/,[a-z]+,/i.match(tempa)

@model=a[0].delete(",")

end

def to\_s

return super+@model

end

end

class Stock<Model

attr\_accessor :stock

def initialize (inputData)

super(inputData)

tempa=inputData.gsub!(/,(\d\*|\d\*\.\d)L\/\d\*km,/i, "")

tempa=inputData.gsub!(@km, "")

tempa=inputData.gsub!(@car\_maker, "")

tempa=inputData.gsub!(@drivetrain, "")

tempa=inputData.gsub!(@trim, "")

tempa=inputData.gsub!(@sets, "")

tempa=inputData.gsub!(@status, "")

tempa=inputData.gsub!(/8L\/10/i, "")

tempa=inputData.gsub!(@type, "")

tempa=inputData.gsub!(@transimission, "")

tempa=inputData.gsub!(@year, "")

tempa=inputData.gsub!(@model, "")

@stock=tempa.gsub!(",", "")

end

def to\_s

return super+@stock

end

end

def convertListings2Catalougue(myList)

inputData=getData(myList)

i=0

myCatalog=[]

while i < inputData.length do

p=Stock.new(inputData[i])

q=p.car\_maker+","+p.model+","+p.trim+","+p.km+","+p.year+","+p.drivetrain+","+p.transimission+","+p.stock+","+p.status+","+p.fuel+","+p.sets

myCatalog<<q

i +=1

end

return myCatalog

end

def searchInventory(myinput)

myinput=myinput.to\_s

a=/=>.\*/i.match(myinput)

i=0

a=a.to\_s

a=a.delete("=>")

a=a.delete("\"")

a=a.delete("}")

File.open("myTXT.txt").each { |line| puts line if line.include? a

}

end

def Add2Inventory(mylist)

temp = File.new("myTXT.txt", "a+")

temp.syswrite("\n")

temp.syswrite(mylist)

mytemplist=","+mylist+","

p=Stock.new(mytemplist)

q=p.car\_maker+","+p.model+","+p.trim+","+p.km+","+p.year+","+p.drivetrain+","+p.transimission+","+p.stock+","+p.status+","+p.fuel+","+p.sets

return q

end

def saveCatalogue2File(mycatalog)

temp=mycatalog.sort

aFile = File.new("final.txt", "a+")

i=0

while i < temp.length do

aFile.syswrite(temp[i])

aFile.syswrite("\n")

i +=1

end

end

puts "mycatalogtest"

myCatalog=convertListings2Catalougue("myTXT.txt")

puts BasicFrame.total

puts myCatalog

puts "--------------------------------------------------"

puts "addtest"

myCatalog<<Add2Inventory("coupe,1100km,auto,RWD,Mercedes,CLK,LX,18FO724A,2017,{AC, Heated Seats, Heated Mirrors, Keyless Entry, Power seats},6L/100km,Used")

puts BasicFrame.total

puts myCatalog

puts "--------------------------------------------------"

puts "savetest,you should check final.txt"

saveCatalogue2File(myCatalog)

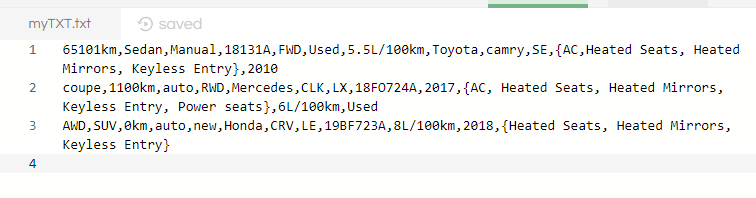
puts "--------------------------------------------------"

puts "searchtest"

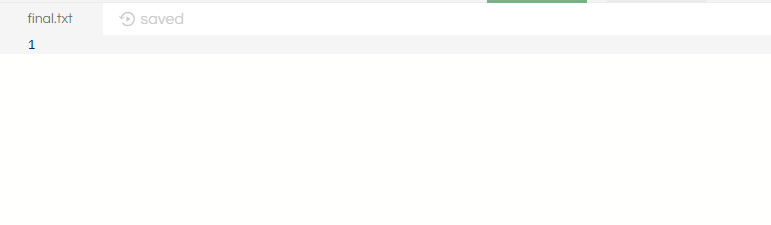
searchInventory ({"car\_maker" => "Mercedes"})

my original txt files

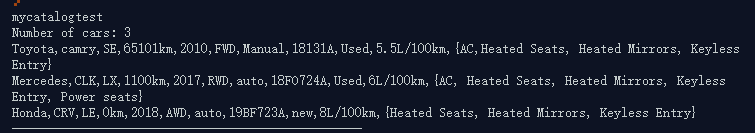
mytxt(input file)



finaltxt(output file)



1. *convertListings2Catalougue*

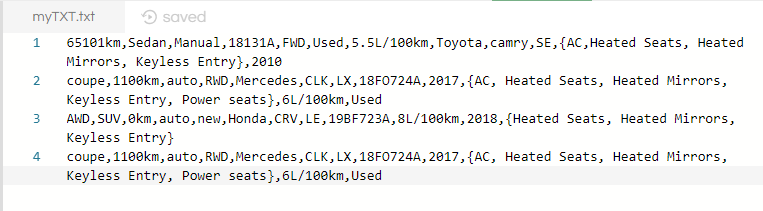


2.search(use medeces here)

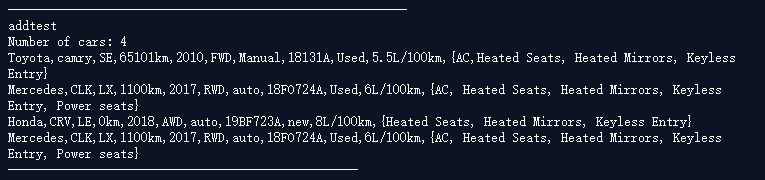


3.add to list (i add a “coupe,1100km,auto,RWD,Mercedes,CLK,LX,18FO724A,2017,{AC, Heated Seats, Heated Mirrors, Keyless Entry, Power seats},6L/100km,Used”)

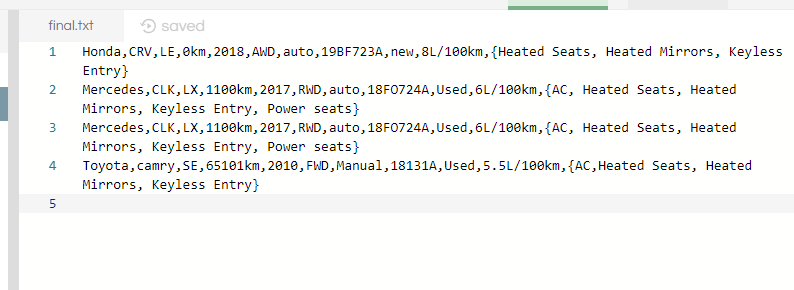
(1)we need to make sure we add a new unordered item into original input txt



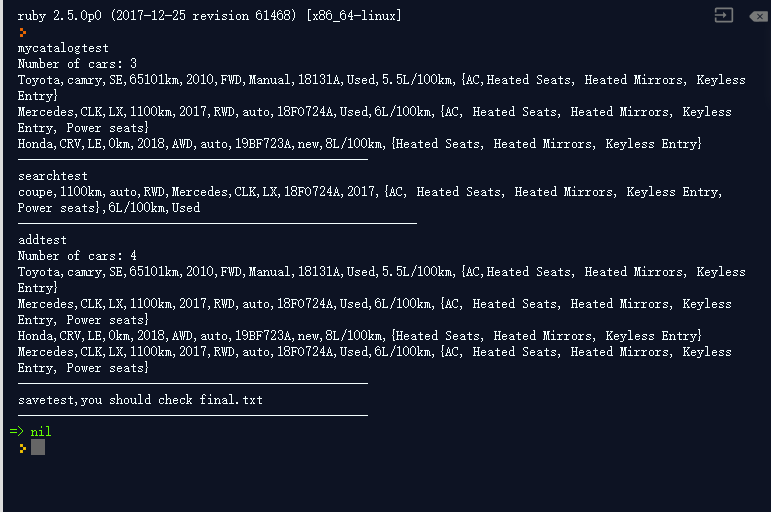
(2) we need to add a order element into my catalog

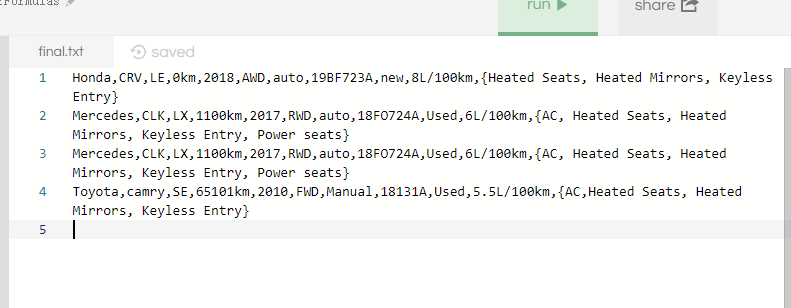


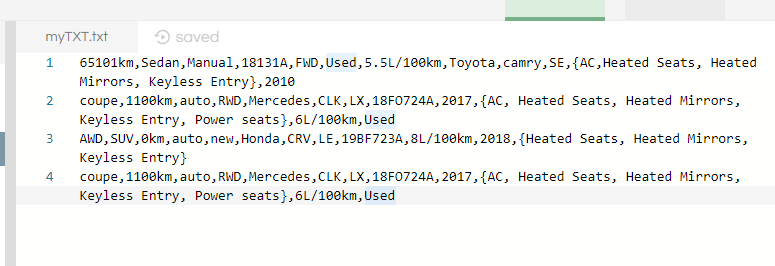
4.save to file (cause we add a new mercedes before, so now we have 4 cars)



total







#include<stdio.h>

void main()

{

int array[2][3],b[3][2];

int i,j;

printf("input:\n");

for(i=0;i<2;i++)

for(j=0;j<3;j++)

{scanf("%d",&array[i][j]);}

printf("\n");

for(i=0;i<2;i++)

{

for(j=0;j<3;j++)

printf("%d ",array[i][j]);

printf("\n");

}

int t;

for(i=0;i<2;i++)

for(j=0;j<3;j++)

{

t=array[i][j];

array[i][j]=b[j][i];

b[j][i]=t;

}

printf("\n");

printf("output");

printf("\n");

printf("\n");

for(i=0;i<3;i++)

{

for(j=0;j<2;j++)

printf("%d ",b[i][j]);

printf("\n");

}

}



5.

#include<stdio.h>

#include<malloc.h>

#include<stdlib.h>

struct car\_maker {

char maker;

struct car\_maker \*next;

struct car\_model \*below;

};

struct car\_model {

char model;

struct car\_model \*next;

struct car\_trim \*below;

};

struct car\_trim {

char trim;

struct car\_trim \*next;

struct car\_km \*below;

};

struct car\_km {

char km;

struct car\_km \*next;

struct car\_year \*below;

};

struct car\_year {

int year;

struct car\_year \*next;

struct car\_type \*below;

};

struct car\_type {

char type;

struct car\_type \*next;

struct car\_transimission \*below;

};

struct car\_transimission {

char transimission;

struct car\_transimission \*next;

struct car\_stock \*below;

};

struct car\_stock {

char stock;

struct car\_stock \*next;

struct car\_status \*below;

};

struct car\_status {

char status;

struct car\_status \*next;

struct car\_fuel \*below;

};

struct car\_fuel {

char fuel;

struct car\_fuel \*next;

struct car\_sets \*below;

};

struct car\_sets {

char sets;

struct car\_sets \*next;

};

int main() {

struct car\_maker \*head = NULL;

struct car\_maker \*new;

head = malloc(sizeof(struct car\_maker));

if (head == NULL)

head->maker = "Honda";

head->next = NULL;

newcar = malloc(sizeof(struct car\_maker));

if (new == NULL) {...}

newcar->maker = "Mercedes";

newcar->next = head;

head = new;

newcar1 = malloc(sizeof(struct car\_maker));

if (newcar1 == NULL) {...}

newcar1->maker = "Toyota";

newcar1->next = head;

head = new;

printf("%d ", head->car\_maker);

printf("%d ", (head->next)->car\_maker);

return 0;

}