Технологично училище “Електронни системи” към
Технически Университет - София

КАК ДА СЕ НАУЧИМ ОТ ГРЕШКИТЕ СИ

Изготвили :

TeamOne

Велислав Костов
Християн Додов
Владимир Йорданов

Дата: 28.10.2014

The first Technology of programming exam took place on the 16th of October. From two classes, totalling 58 people, 34 of them finished their programs, and only 1 of them succeeded on getting a working, finished program(an A). Are the results really terrible or they are normal?

Well, this is our first serious programming exam in school. The time we had was very, very strict and we were downloading and uploading files to other machines. Things not well known for us. We understand the idea behind them, but it'll take some time to get used to them. Nothing is perfect from the first time. On the other side, we had access to internet, books, notebooks. We were free to do whatever we want, the only rule was to not talk to each other, which is a blade with two sides.

We were really confident that it will be easy, when we had access to so much information, that we overestimated the internet and underestimated our problem solving skills. Most of us, knowing we can search the internet for information, refused to use our own thinking. We tried taking different parts of someone else's code, making one bigger program - ours. It was like doing a puzzle, which isn't the fastest and easiest way.

So, there are 15 programs, returning errors, 18 not working as expected and 1 working perfectly. What were the errors? Well, some mistakes were made by more than 1 person. One of the more common ones is not checking for a NilClass. They just have a loop, doing something, calling methods to variables, but they didn't check if there is actually something in this variable. You can't call a method to a not existing object. There were 4 mistakes of this type.

Other type of mistake is not taking advantage of Ruby. Ruby is a high-level language, meaning it does a lot of work instead of you. Some people wrote their code like they are writing C/C++ with Ruby syntax, which makes it way easier to have an error, because it's simply more code.

Some of the people used too much not needed variables, which sooner or later leads to a mistake. There are 2 or 3 errors of this type.

Those are the common errors. They are not that hard to be fixed, so we hope everything will be better the next times. The other not working programs are compiling, but are not doing exactly what it is asked for. This is a serious thing everyone should think about. We have 18 compiling programs and 15 not compiling. Both are not working. Reading the conditions and understanding them well is a huge part of the exam.

The best way to get better results is to write more code. More than just doing homeworks. Even if it's not Ruby, writing code helps you to think like a programmer, which is a big part of this job, hobby, school, or whatever you like to call it. It gives you better problem solving skills, which are crucial. Also, if you get stuck, writing, just think what you want to do, what's your idea. Forget that you must give commands to the machine with code. Imagine that the computer understands bulgarian. What exactly do you want to do? In other words, think about the algorithm, not only the code.

Denis Trenchev[1]:

Problem:

-Misspelled variable names

Fix:

-Fixed variable names

Grade:

-3

Good practice:

-It's a good practice to keep the variable naming conventions of Ruby in mind when writing a Ruby program

Marian Belchev[2]:

Problem:

-Output filename must be exact as mentioned in the task

-Appending numbers after lastname probably mislead by output example due to that append the task's third condition wasn't fulfilled

Fix:

-Removed filename number checking at search and appending at addition to hash which was leading to inaccurate comparing and wrong output

Grade:

-4

Good practice:

-Know what problem you're facing – define your problem. Solve it in minimal amount of steps without extra/unnecessary work.

Ivelin Slavchev[3]:

Problem:

-Tested against fixture with two folders containing files in the correct format (First_Last_digits.rb) and the script outputted all files to the result.csv and according to the task, that's the exact opposite of what should've happened.

Fix:

-No quick fix.

Grade:

-2

Good practice:

-Work on the problem you're solving. It's a good practice to check the data you're working with i.e. check for nil returned by methods.

Stanislav Valkanov[4]:

Problem:

-Design flaw is that only one of two or more files with same firstname will be printed

in result.csv due to hash functionality
-Fixture is probably not accurate, because it contains unformatted files even if the task says that files are in the correct format

Fix:

-Remove invalid files from fixture

Grade:

-3

Good practice:

Petko Bozhinov[5]:

Problem:

-Program probably failed at official fixture due to unformatted files contained in the fixture (which must not be there as written in task)
-Program is outputting single name pair twice (duplicates)
-Found similar code lines{30,35} of current file and
lines {30,33} of Denis Trenchev's file (same functionality, same task)

Fix:

-No quick fix without general code refactoring, using hash will eliminate the duplicates which occur in the output file

Grade:

-3

Good practice:

-Use the most simple approach to a simple task
-Defining/Redefining methods for built in types must be done very carefully and responsibly

Nikola Marinov[6]:

Problem:

-Plenty of syntax errors (related to array/hash use, and general program flow)
-Found similar code lines {30,35} of Petko Bozhinov's file,
lines {30,33} of Denis Trenchev's file (same functionality, same task),
lines {29,32} of current file (same functionality, different task)
-Program uses only one of two passed arguments
-Two almost identical blocks of code (Dir.glob("...").each) which perform similar tasks using only the first argument
-!File was difficult to read, more than difficult to understand and seemed like different blocks of code put together at work which seemed strange

Fix:

-No quick fix,if fix at all
-Consistent use of indexing, and writing programs that have non-linear flow
i.e. don't go straight forward

Grade:

-1

Good practice:

-Firm understanding of Ruby's storage facilities (arrays/hashes etc.), methods related to them,

-Firm understanding of flow control in Ruby

-Avoid syntax errors

Radoslav Kostadinov[7]:

Problem:

-Checking if input was formated when the task guaranteed that it was

-Final comparison between both hashmap elements failed which caused wrong output

Fix:

-Fixed final comparison between the two hashmaps used to store the extracted data

-Removed excess data checking when extracting data from both folders provided as arguments (format check)

Grade:

-3

Good practice:

-Do not overdo the task

-Always try to achieve more with less :)

Simeon Shopkin[8]:

Problem:

-Using String's method 'split' for unformatted string

-Comparing whole array objects instead of their elements (possibly ok in Ruby)

-Using String's method 'split' on Array

Fix:

-No quick fix without general code change

Grade:

-3

Good practice:

-Split strings “safely” and check for nil

-Invoke methods only for objects which type you know

Ivo Valchev[9]:

Problem:

-Performing format checking which wasn't needed

-Minor syntax errors

Fix:

-Removed format checking and fixed syntax errors

Grade:

-2

Good practice:

-Remove format checking ([if 'string'] looks not as pragmatic as [if 'string' != 'empty_string'] even if it works in most cases)

Kristina Pironkova[10]:

Problem:

-Not writing the exact extracted input (raw input) but one with changed case to output file

-Writing output to file results.csv when output is to be read from result.csv

Fix:

-Removed 'capitalize' method because it was changing the working data set (raw input) which was then outputted to result.csv

-Fix output file name

Grade:

-4

Good practice:

-Read task content carefully

-Don't change input if you plan to use it as it was read

Dimitar Nestorov[11]:

Problem:

-Program is crashing at line 42 when trying to sort but the main problem is not there

-Program changes input which may cause invalid comparisons or inaccurate output afterwards

Fix:

-No quick fix, without general code refactoring

Grade:

-2

Good practice:

-Observe the behavior of your programs

Lubomir Yankov[12]:

Problem:

-Task not provided

Fix:

Grade:

-1

Good practice:

-Give chance to anybody that may want to check your work to know what problem you're solving

Tihomir Lidanski[13]:

Problem:

-Found file, but not a program

Fix:

-Nothing to fix

Grade:

-1

Good practice:

Stanislav Gospodinov[14]:

Problem:

-Writing output to results.csv instead of result.csv

Fix:

-Fixed output filename to result.csv

Grade:

-4

Good practice:

Momchil Angelov[15]:

Problem:

-Program crashing

-Code was difficult to read and understand

Fix:

-No quick fix, if fix at all

Grade:

-1

Good practice:

-Make your code readable to yourself and others

Dimitar Terziev[16]:

Problem:

-Program was working without fixes

Fix:

-Nothing to fix

Grade:

-5

Stanislav Iliev [17]:

Problem:

- No quick solution, the code doesn't do what is said by the task.

Fix:

Rate:

Iliyan Germanov [18]:

Problem:

- No apparent problems in the code. Fixture/expected results might be wrong.

Fix:

Rate: 5/5

Stefan Iliev [19]:

Problem:

- Output must be written in "result.csv", not "results.csv".

Fix:

- Csv file name must be changed

Rate: 4/5

Nikolai Mihailov [20]:

Problem:

- Output must be written in "result.csv", not "results.csv".
- Use of the "each" method must be avoided. It doesn't loop through all the characters of a string as expected by the coder. "first.split('').each" could be used.
- "element" must be compared with a string, not an int (for example, "3", not 3).
- No quick solution.

Fix:

Rate:

Borislav Stratev [21] :

Problem:

- Output must be written in "result.csv", not "results.csv".
- The script writes to the csv if only a single filename from the second folder is different from the one compared, not if all the filenames are different, a flag must be added.
- No quick solution.

Fix:

Rate:

Lili Kokalova [22]:

Problem:

- A single filename is written multiple times in the csv file. The second "each" loop must be stopped after a filename is written (line 41).

- No quick solution.

Fix:

Rate:

David Georgiev [23]:

Problem:

- The "length" method must not be used if name[1] is equal to nil.
- There is a quick solution.

Fix:

- Move to next iteration in the "Dir.glob" loop if the name is nil.

Rate: 5/5

Valentin Varbanov [24]:

Problem:

- Invalid values of name[1] and name[2] are being pushed in the array (line 57). A checker must be added to make sure nothing is nil.
- No quick solution.

Fix:

Rate:

Veselina Kolova [25]:

Problem:

- Syntactically wrong "if" with many unnecessary checks (line 23).
- Names are written in the wrong order.
- Sorting doesn't work.
- There is a quick solution.

Fix:

- Remove the unneeded checks in the "if" statement (line 23).
- Write first name then last name, not last name then first name.
- Use another method for the sorting.

Rate: 4/5

Appendixes:

[1] -

=begin

Develop a program named FirstName_LastName_ClassNumber_b4c3f5.rb

1. you are given two arguments for a folders with files;
1.1 if there are other arguments they should be discarded;
2. file names in this folders are in the form First_Last_digits.rb;
3. find the students with 5 letters in the first name that are in both folders. A student is in both folders if it there is a file with the same First and Last Name. Digits might be different;
4. Sort the result by Last name ;
5. Produce a result in CSV format named result.csv:

```
	LastName1,FirstName1
	LastName2,FirstName2
	...
	LastNameN,FirstNameN

=end

require 'csv'

i = 0
arr1 = []
arr2 = []
arr3 = []

Dir.glob(ARGV[0]+"*.rb") do |first_folder|
	name = first_folder.split('/').last.split('.').first.split('_')

	if name.length == 3
		if name[0].to_s.length == 5
			arr1[i] = []
			arr1[i][0] = name[0]
			arr1[i][1] = name[1]
			i+=1
		end
	end
end
i = 0

Dir.glob(ARGV[1]+"*.rb") do |second_folder|
	name = second_folder.split('/').last.split('.').first.split('_')

	if name.length == 3
		if name[0].to_s.length == 5
			arr2[i] = []
			arr2[i][0] = name[0]
```

```
                        arr2[i][1] = name[1]
                        i+=1
                end
        end
end
i = 0

arr1.each do |compare1|
        arr2.each do |compare2|
                if compare2 == compare1
                        arr3[i] = compare1
                        i+=1
                end
        end
end

sort = arr3.sort_by{|asd| asd[1]}
CSV.open("result.csv", "w") do |csv|
   sort.each do |element|
      csv << element
   end
end
```

[2] -

```
        =begin
Develop a program named FirstName_LastName_ClassNumber_ad26e0.rb

1. you are given two arguments for a folders with files;
1.1 if there are other arguments they should be discarded;
2. file names in this folders are in the form First_Last_digits.rb;
3. find  the students that are only in the second folder and not in the first. A student is in both
folders if it there is a file with the same First and Last Name. Digits might be different;
4. Sort the result by First name ;
5. Produce a result in CSV format named result.csv:

        LastName1,FirstName1
        LastName2,FirstName2
        ...
        LastNameN,FirstNameN
=end
```

```
require 'csv'

hash1 = Hash.new
hash2 = Hash.new

Dir.glob("#{ARGV[0]}*_*_*.rb") do |file1|
	Dir.glob("#{ARGV[1]}*_*_*.rb") do |file2|
		firstName1	= file1.split("/").last.split("_").first
		lastName1	= file1.split("/").last.split("_")[1]

		firstName2	= file2.split("/").last.split("_").first
		lastName2	= file2.split("/").last.split("_")[1]

		hash1[firstName1] = lastName1 #+ "." + number1
		hash2[firstName2] = lastName2 #+ "." + number2
	end
end
CSV.open("result.csv", "w") do |csv|
	hash2.sort.each do |key, value|
		if hash1[key] != value
			csv << [key,value.gsub('.',"")]
		end
		#if hash1.has_key?(key) && !hash1.has_value?(value)
		#	csv << [key,value.gsub('.',"")]
		#end
	end
end
```

[3]-

	=begin
	Develop a program named FirstName_LastName_ClassNumber_835552.rb

1. you are given two arguments for a folders with files;
1.1 if there are other arguments they should be discarded;
2. Find all the files from both folders that are not in the format FirsrName_LastName_digits.rb. If there are duplicates the file must be written only once. If two files are of the same lenght those files should be sorted in ASC order;
3. Calculate the length of their names (including extensions).;
4. Sort the result by lenth ;
5. Produce a result in CSV format named result.csv:

File1,3
File2,4

```
                        ...
                        FileN,3
=end

require 'csv'
result = Hash.new
Dir.glob(ARGV[0] + "*").each do |file1|
        short1 = file1.split("/").last
        ext1 = short1.split(".").last
        names1 = short1.split(".").first
        digit1 = file1.split("_").last
        if ((ext1 != "rb") or (digit1.to_i.to_s != digit1) or (short1.scan("_").count < 2))
                result[short1] = short1.length
        end
end
Dir.glob(ARGV[1] + "*").each do |file2|
        short2 = file2.split("/").last
        ext2 = short2.split(".").last
        names2 = short2.split(".").first
        digit2 = file2.split("_").last
        if ((ext2 != "rb") or (digit2.to_i.to_s != digit2) or (short2.scan("_").count < 2))
                result[short2] = short2.length
        end
end
result.sort_by{|k, v| v}
CSV.open("result.csv", "w") do |csv|
        result.each do |p|
                csv << p
        end
end
```

[4]-

```
        #Develop a program named FirstName_LastName_ClassNumber_4482c1.rb

#1. you are given an argument for a folder with files;
#1.1 if there are other arguments they should be discarded
#2. file names in this folder are in the form First_Last_digits.rb;
#3. find all the students that have 5 letters in their second name;
#4. Sort the result by First name DESC.
#5. Produce a result in CSV format named result.csv:

#               FirstName1,LastName1
```

```
#               FirstName2,LastName2
#               ...
#               FirstNameN,LastNameN

require 'csv'
a = Hash.new
path = ARGV[0]
Dir.glob(path + "**/*.rb") do |my_text_file|
short_name = my_text_file.split('/').last.split('.').first
name = short_name.split("_")[0]
last = short_name.split("_")[1]
last.to_s
if (last.length == 5)&&(short_name.split("_").size == 3)
a["#{name}"] = last
end
end
CSV.open("result.csv", "w") do |csv|
Hash[a.sort.reverse].each do |element|
csv << element
end
end
```

[5]-

```
        # Develop a program named FirstName_LastName_ClassNumber_954dc6.rb

# 1. you are given two arguments for a folders with files;
# 1.1 if there are other arguments they should be discarded;
# 2. file names in this folders are in the form First_Last_digits.rb;
# 3. find  the students with 5 letters in the first name that are in both folders. A student is in both
folders if it there is a file with the same First and Last Name. Digits might be different;
# 4. Sort the result by Last name ;
# 5. Produce a result in CSV format named result.csv:

#       LastName1,FirstName1
#       LastName2,FirstName2
#       ...
#       LastNameN,FirstNameN

require 'csv'

class String
  def numeric?
    Float(self) != nil rescue false
```

```
  end
end

output = Array.new
i = 0
Dir.glob(ARGV[0] + "/*") do |file|
	file = file.split('/').last.split('.').first.split('_')
	Dir.glob(ARGV[1] + "/*") do |file2|
		file2 = file2.split('/').last.split('.').first.split('_')
		if "#{file[0]} #{file[1]}" == "#{file2[0]} #{file2[1]}"
			if file[2].numeric?
				if file[0].to_s.length == 5
					output[i] = Array.new
					output[i][0] = file[0]
					output[i][1] = file[1]
					i+=1
				end
			end
		end
	end
end

output = output.sort_by{ |element| element[1]}
CSV.open("result.csv", "w") do |csv|
	output.each do |pusher|
		csv << pusher
	end
end
```

[6]-

=begin

1. you are given two arguments for a folders with files;
1.1 if there are other arguments they should be discarded;
2. Find all the files from both folders that have exactly 7 digits from 0 to 9 in their names excluding extension. If there are duplicates the file must be written only once.;
3. Calculate the length of their names (including extensions) divided by 2 rounded to the smalles number;
4. Sort the result by File name ;
5. Produce a result in CSV format named result.csv:

```
File1,3
File2,4
...
```

```
                              FileN,3
=end

require 'csv'
def is_numeric(o)
 true if Integer(o) rescue false
 end

 array=[]

count=0
Dir.glob(ARGV[0] + "/**/*.*").each do |file|

 full_name=file.split("/").last
 name = file.split("/").last.split(".").first_split("_")

 if name.lenght != 3 && !is_numeric(name[2])
 array[count] = []
 array[count][0]=full_name
 array[count][1]= full_name.to_s.lenght
 count += 1

 end
 end

 Dir.glob(ARGV[0] + "/**/*.*").each do |file|

 full_name=file.split("/").last
 name = file.split("/").last.split(".").first_split("_")

 if name.lenght != 3 && !is_numeric(name[2])
 array[count] = []
 array[count][0]=full_name
 array[count][1]= full_name.to_s.lenght
 count += 1
 end
 end
 array = array.sort_by{|el| el[0]}

 CSV.open("task.csv","w") do |csv|
 array=uniq.each do |element|
 csv << element
 end
```

```
 end
```

[7]-

```
        =begin
Develop a program named FirstName_LastName_ClassNumber_772118.rb

1. you are given two arguments for a folders with files;
1.1 if there are other arguments they should be discarded;
2. file names in this folders are in the form First_Last_digits.rb;
3. find  the students that are only in the second folder and not in the first. A student is in both
folders if it there is a file with the same First and Last Name. Digits might be different;
4. Sort the result by First name ;
5. Produce a result in CSV format named result.csv:

        LastName1,FirstName1
        LastName2,FirstName2
        ...
        LastNameN,FirstNameN

=end

require 'csv'
file1 = Hash.new
file2 = Hash.new

path1 = ARGV[0]
path2 = ARGV[1]

Dir.glob("#{path1}*.rb") do |my_text_file|
        first_name = my_text_file.split("/").last.split("_").first
        last_name  = my_text_file.split("/").last.split("_")[1]
   file1[first_name] = last_name
end

Dir.glob("#{path2}*.rb") do |my_text_file|
                first_name = my_text_file.split("/").last.split("_").first
                last_name  = my_text_file.split("/").last.split("_")[1]
                file2[first_name] = last_name
end

CSV.open("result.csv", "w") do |csv|
        file2.each do |first_name, last_name|
                if (file1[first_name] <=> last_name) != 0
```

```
                        csv << [last_name, first_name]
                end
        end
end
```

[8]-

=begin

Develop a program named FirstName_LastName_ClassNumber_56a835.rb

1. you are given two arguments for a folders with files;
1.1 if there are other arguments they should be discarded;
2. Find all the files from both folders that are not in the format FirsrName_LastName_digit.rb. If there are duplicates the file must be written only once. If two files are of the same lenght those files should be sorted in ASC order;
3. Calculate the length of their names (including extensions).;
4. Sort the result by length ;
5. Produce a result in CSV format named result.csv:

```
File1,3
File2,4
...
FileN,3
```

=end

```
require 'csv'

arr = Array.new
        Dir.glob(ARGV[0]+"/*.rb") do |first_files|
                Dir.glob(ARGV[1]+"/*.rb") do |second_files|
                        first_files = first_files.split("/").last.split(".").first.split("_")
                        if first_files.size != 3
                                if first_files != second_files
                                        print_count = first_files.split("/").last.split(".").first
                                        p = print_count.size.to_s
                                        print =
first_files[0].capitalize+"_"+first_files[1].capitalize+"_"+first_files[2]+","+p
                                        arr.push(print)
                                end
                        end
                end
        end
```

```
	CSV.open("result.csv","w") do |csv|
		arr.sort.each do |element|
			csv << [element]
		end
	end
```

[9]-

```
	=begin
Develop a program named FirstName_LastName_ClassNumber_6c8bd9.rb

1. you are given two arguments for a folders with files;
1.1 if there are other arguments they should be discarded;
2. file names in this folders are in the form First_Last_digits.rb;
3. find  the students with 5 letters in the first name that are in both folders. A student is in both
folders if it there is a file with the same First and Last Name. Digits might be different;
4. Sort the result by Last name ;
5. Produce a result in CSV format named result.csv:

	LastName1,FirstName1
	LastName2,FirstName2
	...
	LastNameN,FirstNameN
=end

hash_fold1={}
hash_fold2={}

Dir.glob("#{ARGV[0]}*.*")  do |file|
		name = file.split("/").last.split(".").first.split("_")
		if name[0].length == 5 and !hash_fold1.include?(name[0])
			hash_fold1["#{name[1]}"] = "#{name[0]}"
		end
end
Dir.glob("#{ARGV[1]}*.*") do |file|
		name = file.split("/").last.split(".").first.split("_")
		if name[0].length == 5 and !hash_fold2.include?(name[0])
			hash_fold2["#{name[1]}"] = "#{name[0]}"
		end
end
File.open("result.csv", "w") do |csv|
	hash_fold1.sort.map do |key, value|
		if (hash_fold1[key] == hash_fold2[key])
			csv.puts("#{key},#{value}")
```

```
        end
    end
end
```

[10]-

```
    =begin
Develop a program named FirstName_LastName_ClassNumber_890ba0.rb

1. you are given an argument for a folder with files;
1.1 if there are other arguments they should be discarded
2. file names in this folder are in the form First_Last_digits.rb;
3. find all the students that have 10 letters in their first name;
4. Sort the result by Last Name DESC.
5. Produce a result in CSV format named result.csv:

        FirstName1,LastName1
        FirstName2,LastName2
        ...
        FirstNameN,LastNameN
=end


require 'csv'
results=Hash.new
Directory = ARGV[0]
Dir.glob("#{Directory}/*.rb") do |file_name|

    first_name = file_name.split("/").last.split("_").first
    last_name=file_name.split("/").last.split("_",2).last.split("_").first

        if first_name.length == 10

            results["#{last_name}"] ="#{first_name}"
        end

end


CSV.open("result.csv", "w") do |csv|
    results.sort.each do |first,last|

    csv << [last,first]
```

```
	end
end
```

[11]-

```
	#Develop a program named FirstName_LastName_ClassNumber_0d5526.rb
#
#1. you are given an argument for a folder with files;
#1.1 if there are other arguments they should be discarded
#2. file names in this folder are in the form First_Last_digits.rb;
#3. find all the students that have 10 letters in their first name;
#4. Sort the result by Last Name DESC.
#5. Produce a result in CSV format named result.csv:
#
#		FirstName1,LastName1
#		FirstName2,LastName2
#		...
#		FirstNameN,LastNameN


require 'csv'
def is_numeric(o)
   true if Integer(o) rescue false
end
array = []
count = 0
Dir.glob(ARGV[0] + "*.rb") do |file|
	name = file.split("/").last.split(".").first.split("_")

	name[0] = name[0].to_s
	name[0] = name[0].capitalize

	name[1] = name[1].to_s
	name[1] = name[1].capitalize

	if name.size == 3 && is_numeric(name[2])
		if name[1].length == 10

			array[count] = []
			array[count][0] = name[0].to_s
			array[count][1] = " #{name[1].to_s}"
			count += 1

		end
```

```
        end
end
array = array.sort_by {|el| -zel[1]}
CSV.open("result.csv", "w") do |csv|

                array.uniq.each do |e|

                        csv << e

                end

end
```

[12]-

```
        require 'csv'
def is_numeric(o)
   true if Integer(o) rescue false
end

array = []
count = 0

Dir.glob(ARGV[0] + "*").each do |file|
        ch_count = 0
        file_name = file.split("/").last.split("")

        file_name.each do |ch|

                if is_numeric(ch)

                        ch_count += 1

                end

        end
        if ch_count == 9
                len = file_name.length
                array[count] = []
                array[count][0] = file_name
                array[count][1] = len/2.round
                count += 1
        end
```

```
end
array = array.sort_by {|el| el[0]}
CSV.open("result.csv", "w") do |csv|

        array.each do |element|

                csv << element

        end

end
```

[13]-

```
        #Develop a program named FirstName_LastName_ClassNumber_dafd44.rb

#1. you are given two arguments for a folders with files;
#1.1 if there are other arguments they should be discarded;
#2. Find all the files from both folders that have exactly 7 digits from 0 to 9 in their names excluding extension. If there are duplicates the file must be written only once.;
#3. Calculate the length of their names (including extensions) divided by 2 rounded to the smalles number;
#4. Sort the result by File name ;
#5. Produce a result in CSV format named result.csv:

#                       File1,3
#                       File2,4
#                       ...
#                       FileN,3

require 'csv'

Dir.glob(ARGV[0] + "*.") do |file|
        name = file.split ("/")last.split(".")


Dir.glob(ARGV[1] + "*.") do |file|




puts name.length % 2.round()
```

```
end
end

CSV.open("result.csv", "w") do |csv|



end
```

[14]-

```
	=begin
Develop a program named FirstName_LastName_ClassNumber_b36abb.rb

1. you are given an argument for a folder with files;
1.1 if there are other arguments they should be discarded
2. file names in this folder are in the form First_Last_digits.rb;
3. find all the students that have 5 letters in their second name;
4. Sort the result by Last Name ASC.
5. Produce a result in CSV format named result.csv:

		FirstName1,LastName1
		FirstName2,LastName2
		...
		FirstNameN,LastNameN
=end

require 'csv'
hash = Hash.new

Dir.glob("#{ARGV[0]}*.rb") do |file|
	filename = file.split('/').last.split('.').first;
		if filename.split('_').length == 3
			if filename.split('_')[1].length == 5
				hash[filename.split('_')[0]] = filename.split('_')[1]
			end
		end
end

hash = Hash[hash.sort_by{|k, v| v}]

CSV.open("result.csv", "w") do |csv|
	hash.each do |key, value|
```

```
            csv << [key, value].flatten
        end
end
```
[15]-
```
        =begin
```
Develop a program named FirstName_LastName_ClassNumber_d8aa65.rb

1. you are given two arguments for a folders with files;
1.1 If there are other arguments they should be discarded;
2. Find all the files from both folders that are not in the format FirsrName_LastName_digits.rb. If there are duplicates the file must be written only once.
2.1 If two files are of the same lenght those files should be sorted in ASC order;
3. Calculate the length of their names (including extensions).;
4. Sort the result by lenth ;
5. Produce a result in CSV format named result.csv:

File1,3
File2,4
...
FileN,3

```
=end
require 'csv'

arr1=Array.new
arr2=Array.new
arr3=Array.new
a = ARGV[0]
b = ARGV[1]
i=0
Dir.glob(a + "/*.rb") do |my_text_file1|
        short= my_text_file1.split('/').last
        length1 = short.length
        shorter= short.split('.').first.split('_')
        first_name=shorter[0]
        last_name=shorter[1]
        digits=shorter[2].to_i


        if !first_name || !last_name || digits=0
                next
        else
                arr1 << ["#{short}" "#{length1}"]
        end
```

```
end
Dir.glob(b + "/*.rb") do |my_text_file2|

        short2= my_text_file2.split('/').last
        length2 = short2.length
        shorter2= short2.split('.').first.split('_')
        first_name2=shorter2[0]
        last_name2=shorter2[1]
        digits2=shorter2[2].to_i


        if !first_name2 || !last_name2 || digits2=0
                next
        else
                arr2 << ["#{short2}","#{length2}"]
        end
end

        arr3 = arr1 & arr2



        arr3 = arr3.sort_by  {|el|
                 el[1]
        }



    CSV.open("result.csv", "w") do |csv|

arr3.each do |element|
csv << element
end

end
```

[16]-

```
        =begin
Develop a program named FirstName_LastName_ClassNumber_88db52.rb

1. you are given an argument for a folder with files;
1.1 if there are other arguments they should be discarded
2. file names in this folder are in the form First_Last_digits.rb;
```

3. find all the students that have 5 letters in their second name;
4. Sort the result by Last Name ASC.
5. Produce a result in CSV format named result.csv:

```
FirstName1,LastName1
FirstName2,LastName2
...
FirstNameN,LastNameN
```

```
=end
require 'csv'
arr = []
Dir.glob("#{ARGV[0]}*.rb*"){|file|
	file_str = file.split('/').last
	if(file_str=~/\A[a-zA-Z]+\_[a-zA-Z]+\_\d+\.rb\z/ && file_str.split('_')[1].size == 5)
		arr.push("#{file_str.split('_')[1]} #{file_str.split('_').first}")
	end
}
CSV.open('result.csv','w'){|csv|
	arr.uniq.sort.each{|el|
		csv << "#{el.split(' ').last} #{el.split(' ').first}".split(' ')
	}
}
```

[17]

```
#Develop a program named FirstName_LastName_ClassNumber_627d43.r#
#
#1. you are given two arguments for a folders with files;
#1.1 if there are other arguments they should be discarded;
#2. file names in this folders are in the form First_Last_digits.rb;
#3. find  the students that are only in the first folder and not in the second. A student is in both
folders if it there is a file with the same First and Last #Name. Digits might be different;
#4. Sort the result by Last name ;
#5. Produce a result in CSV format named result.csv:
#
#	LastName1,FirstName1
#	LastName2,FirstName2
#	...
#	LastNameN,FirstNameN
```

```
require 'csv'
name_array = Array.new()
name_array2 = Array.new()
support_array = Array.new()
support_array2 = Array.new()
i = 0
dir1 = ARGV[0]
dir2= ARGV[1]

Dir.glob("#{dir1}/*.*") do |file|
        name_array[i] = file.split(/\//).last
        i += 1
end
count = i
i = 0
Dir.glob("#{dir2}/*.*") do |file2|
        name_array2[i] = file2.split(/\//).last
        i += 1
end
i = 0
for check in i..count
        if name_array[check] != name_array2[check]
                        support_array[i] = name_array[check]
                        support_array2[i] = name_array2[check]
                        i += 1
                        puts support_array
                        puts support_array2
                        CSV.open("result.csv", "w") do |csv|
                                support_array.each do |element|
                                        csv << [element]
                                end
                        end
                        CSV.open("result.csv", "w") do |csv|
                                support_array2.each do |element2|
                                                csv << [element2]
                                end
                        end
        end
end
```

[18]

```
=begin
        Develop a program named FirstName_LastName_ClassNumber_f8b0d9.rb

1. you are given two arguments for a folders with files;
1.1 if there are other arguments they should be discarded;
2. file names in this folders are in the form First_Last_digits.rb
3. find  the students that are only in the first folder and not in the second. A student is in both
folders if it there is a file with the same First and Last Name. Digits might be different;
4. Sort the result by Last name ;
5. Produce a result in CSV format named result.csv:

        LastName1,FirstName1
        LastName2,FirstName2
        ...
        LastNameN,FirstNameN
=end

require 'csv'
results = Hash.new
results.compare_by_identity
def is_number(str)
        str[/[0-9]+/] == str
end
Dir.glob("#{ARGV[0]}/*.rb") do |path1|
        filename1 = path1.split(/\//).last
        if filename1.count("_") == 2
                firstname1 = filename1.split("_").first
                lastname1 = filename1.split("_")[1]
                digit1 = filename1.split("_")[2].split(".").first
                if is_number(digit1)
                        flag = 0
                        Dir.glob("#{ARGV[1]}/*.rb") do |path2|
                                filename2 = path2.split(/\//).last
                                if filename2.count("_") == 2
                                        digit2 = filename2.split("_")[2].split(".").first
                                        if is_number(digit2)
                                                name1 = firstname1 + lastname1
                                                name2 = filename2.split("_").first +
filename2.split("_")[1]
                                                if name1 == name2
                                                        flag = 1
```

```
                                                        break
                                                end
                                        end
                                end
                        end
                        if flag == 0
                                results[lastname1] = firstname1
                        end
                end
        end
end

CSV.open("result.csv", "w") do |csv|
        results.sort_by{|key, val| key}.each do |el|
                csv << el
        end
end
```

[19]

```
#Develop a program named FirstName_LastName_ClassNumber_d77aee.rb
#
#1. you are given two arguments for a folders with files;
#1.1 if there are other arguments they should be discarded;
#2. Find all the files from both folders that are not in the format FirsrName_LastName_digit.rb. If there are duplicates the file #must be written only once. If two files are of the same lenght those files should be sorted in ASC order;
#3. Calculate the length of their names (including extensions).;
#4. Sort the result by length ;
#5. Produce a result in CSV format named result.csv:
#
#                       File1,3
#                       File2,4
#                       ...
#                       FileN,3

require 'csv'

first_folder = ARGV.shift
second_folder = ARGV.shift || "err"
names_hash = Hash.new

Dir.glob(first_folder+"/*.*").each do |text_file|
        text_file = text_file.split("/").last
```

```
	if (text_file.split("_").length == 3) then
		first_name = text_file.split("_")[0]
		second_name = text_file.split("_")[1]
		diggit = text_file.split("_")[2].split(/\./).first
		if (diggit.to_i.to_s != diggit) then names_hash[text_file] = text_file.length end
		if (first_name =~ /\d/) then names_hash[text_file] = text_file.length end
		if (second_name =~ /\d/) then names_hash[text_file] = text_file.length end
	else
		names_hash[text_file] = text_file.length
	end
end

if second_folder != "err"
	Dir.glob(second_folder+"/*.*").each do |text_file|
		text_file = text_file.split("/").last
		if (text_file.split("_").length == 3) then
			first_name = text_file.split("_")[0]
			second_name = text_file.split("_")[1]
			diggit = text_file.split("_")[2].split(/\./).first
			if (diggit.to_i.to_s != diggit) then names_hash[text_file] = text_file.length
end
			if (first_name =~ /\d/) then names_hash[text_file] = text_file.length end
			if (second_name =~ /\d/) then names_hash[text_file] = text_file.length end
		else
			names_hash[text_file] = text_file.length
		end
	end
end

names_hash = Hash[names_hash.sort_by{|k,v| k} ]
names_hash = Hash[names_hash.sort_by{|k,v| v} ]

puts names_hash

CSV.open("results.csv","w") do |csv|
	names_hash.each do |element|
		csv << element
	end
end
```

[20]

#Develop a program named FirstName_LastName_ClassNumber_f70059.rb

#1. you are given two arguments for a folders with files;
#1.1 if there are other arguments they should be discarded;
#2. Find all the files from both folders that have exactly 7 digits from 0 to 9 in their names excluding extension. If there are duplicates the file must be written only once.;
#3. Calculate the length of their names (including extensions) divided by 2 rounded to the smalles number;
#4. Sort the result by File name ;
#5. Produce a result in CSV format named result.csv:

```
#                       File1,3
#                       File2,4
#                       ...
#                       FileN,3

require 'csv'
hash = Hash.new
count = 0
        Dir.glob(ARGV[0] + "/*.rb") do |file|

                first = file.split(/\//).last
                puts first

                #for (i = 0;i < first.length;i+=1)
                size = first.length
                i = 0
                first.each do |element|

                        c = first[i].chr
                        if element == 0 || element == 1 || element == 2 || element == 3 || element
== 4 || element == 5 || element == 6 || element == 7 || element == 8 || element == 9
                        count +=1

                        end
                end
                puts count
        end

        Dir.glob(ARGV[1] +"/*.rb") do |secFile|
                sec = secFile.split(/\//).last
                #puts sec

        end
```

```
CSV.open("result.csv", "w") do |csv|
        hash.sort_by{|key,val| key}.each do |element|
        csv << element
        end
end
```

[21]

```
#Develop a program named FirstName_LastName_ClassNumber_a65be5.rb

#1. you are given two arguments for a folders with files;
#1.1 if there are other arguments they should be discarded;
#2. file names in this folders are in the form First_Last_digits.rb;
#3. find  the students that are only in the first folder and not in the second. A student is in both
folders if it there is a file with the same First and Last Name. Digits might be different;
#4. Sort the result by Last name ;
#5. Produce a result in CSV format named result.csv:

#       LastName1,FirstName1
#       LastName2,FirstName2
#       ...
#       LastNameN,FirstNameN

require 'csv'
a = Array.new
h = Hash.new
Dir.glob("#{ARGV[0]}/*.rb") do |dir_file_name_1|
        Dir.glob("#{ARGV[1]}/*.rb") do |dir_file_name_2|

                file_name_1 = dir_file_name_1.split(/\//).last.to_s
                file_name_2 = dir_file_name_2.split(/\//).last.to_s

                if(file_name_1 != file_name_2)
                        file_name = file_name_1
                        digit = file_name.split(/_/).last.split(/\./).first.to_s
                        first_name = file_name.split(/_/).first.to_s
                        full_first_name = first_name + digit
                        full_first_name = full_first_name.to_s
                        tmp = file_name.split("#{first_name}_")
                        full_last_name = tmp.last.split(/_/).first.to_s + digit
                        full_last_name = full_last_name.to_s
                        h[full_last_name] = full_first_name
```

```
                end
        end
end

CSV.open("results.csv", "w") do |csv|
        a = h.sort
        a.each do |element|
                csv << element
        end
end
```

[22]

```
=begin
Develop a program named FirstName_LastName_ClassNumber_e0ea9c.rb

1. you are given two arguments for a folders with files;
1.1 if there are other arguments they should be discarded;
2. file names in this folders are in the form First_Last_digits.rb;
3. find  the students that are only in the second folder and not in the first. A student is in both
folders if it there is a file with the same First and Last Name. Digits might be different;
4. Sort the result by First name ;
5. Produce a result in CSV format named result.csv:

        LastName1,FirstName1
        LastName2,FirstName2
        ...
        LastNameN,FirstNameN
=end

require 'csv'
student = Array.new
student1 = Array.new

Dir.glob(ARGV[0]+"/**/*.*").each do |file_name1|
        file_name = file_name1.split("/").last
        first_name = file_name.split("/").last.split("_").first
        p first_name
        last_name = file_name.split("/").last.split("_",2).last.split("_").first
        #task = file_name.split("_").last.split(".").first
        student << ["#{first_name}", "#{last_name}"]
end
```

```
Dir.glob(ARGV[1]+"/**/*.*").each do  |file_name1|
	file_name = file_name1.split("/").last
	first_name = file_name.split("/").last.split("_").first
	p first_name
	last_name = file_name.split("/").last.split("_",2).last.split("_").first
	#task = file_name.split("_").last.split(".").first
	student1 << ["#{first_name}", "#{last_name}"]
end

CSV.open("result.csv", "w") do |csv|
	student.each do |fn, ln|
		student1.each do |fn1, ln1|
			if fn != fn1
				if ln != ln1
					csv << ["#{fn1}", "#{ln1}"]
				end
			end
		end
	end
end
```

[23]

```
#Develop a program named FirstName_LastName_ClassNumber_1eea4f.rb
#1. you are given an argument for a folder with files;
#1.1 if there are other arguments they should be discarded
#2. file names in this folder are in the form First_Last_digits.rb;
#3. find all the students that have 5 letters in their second name;
#4. Sort the result by Last Name ASC.
#5. Produce a result in CSV format named result.csv:
# FirstName1,LastName1
# FirstName2,LastName2
# ...
# FirstNameN,LastNameN

require 'csv'
students_names = []

Dir.glob("#{ARGV[0]}/**/*.rb") do |current_file|
	name = current_file.split('/').last.split(/_/)
	next if name[1] == nil # => added line
	if name[1].length == 5
		if not students_names.include?(["#{name[1]}", "#{name[0]}"]) then
```

```
                        students_names << (["#{name[1]}", "#{name[0]}"])
                end
        end
end

CSV.open("result.csv", "w") do |csv|
        students_names.sort.each do |last, first|
                csv << ["#{first}", "#{last}"]
        end
end
```

[24]

```
=begin

Develop a program named FirstName_LastName_ClassNumber_041472.rb

1. you are given two arguments for a folders with files;
1.1 if there are other arguments they should be discarded;
2. file names in this folders are in the form First_Last_digits.rb;
3. find  the students that are only in the first folder and not in the second. A student is in both
folders if it there is a file with the same First and Last Name. Digits might be different;
4. Sort the result by Last name ;
5. Produce a result in CSV format named result.csv:

        LastName1,FirstName1
        LastName2,FirstName2
        ...
        LastNameN,FirstNameN


=end

students_first_dir = Array.new
students_second_dir = Array.new

for i in 0..1

        directory = ARGV[i]
        if ARGV[i].split(//).last(1).to_s == "/"
                directory += "**/*.rb"
        else
                directory += "/**/*.rb"
        end
```

```
	Dir.glob(directory).each do |dir|
		student = dir.split(/\//)
		if i == 0
			students_first_dir.push(student)
		else
			students_second_dir.push(student)
		end
	end
end

studentcsv = Array.new

students_first_dir.each do |std|
	match = 0
	students_second_dir.each do |std2|
		name = std.last.split(/_/)

		name2 = std2.last.split(/_/)
		for i in 0..1
			if name[i] == name2[i]
				match = 1
			end
		end

	end
	studentcsv.push(name[1], name[2])
end

CSV.open("result.csv", "w") do |csv|
	studentcsv.each do |string|
		csv << string
	end
end
```

[25]

```
=begin
Develop a program named FirstName_LastName_ClassNumber_65630e.rb
1. you are given an argument for a folder with files;
1.1 if there are other arguments they should be discarded
2. file names in this folder are in the form First_Last_digits.rb;
3. find all the students that have 5 letters in their second name;
4. Sort the result by First name DESC.
```

5. Produce a result in CSV format named result.csv:
FirstName1,LastName1
FirstName2,LastName2
...
FirstNameN,LastNameN
=end

```
require 'csv'

people = Hash.new

Dir.glob("#{ARGV[0]}/**/*.*").each do |text_file|

        if File.extname(text_file) then # => modified line
#old => if File.extname(text_file) text_file.include?(".rb") &&
text_file.split(/_/).last.split(/\./).first.to_i.is_a Integer then
                if (text_file.split("/").last.split("_").length == 3) then
                        text_file = text_file.split("/").last
                        if (text_file.split("_")[1].length == 5) then
                                people[text_file.split("_")[0]] = text_file.split("_")[1] # => modified
line
                                #old => people[text_file.split("_")[1]] = text_file.split("_")[0]
                        end
                end
        end
end

people = people.sort.reverse # => modified line
#old => people = Hash[people.sort_by{|k,v| k}.reverse]

CSV.open("result.csv","w") do |csv|
        people.each do |element|
        csv << element
        end
end
```