Технологично училище Електронни Системи

към Технически Университет - София

Дата: октомври 2014

Team: require ‘teamName’

Ивелин Славчев

Красен Ангелов

Любомир Янков

- The most common problem is that the output CSV file name of the program is incorrect.
- In most cases the output CSV file is named “results.csv” instead of “result.csv”.
- Another mistake is that most students are missing a checking statement for correct file name format .
- There are also a lot of syntax errors in the programs, such as undefined methods, mistaken variable names and wrong statements.
- The number of errors in most programs is between 1 to 3. Some of them are easily fixable, but others don’t compile.
- Some students aren’t even following the task description.
- There are some programs that would output correct results if not for a few minor incorrections, but there are others that have major errors.
- Most of these program errors can be avoided by simply paying more attention to the given task, thoroughly looking through the code, before submitting and practicing more at home.
- Some errors are caused by the usage of complicated statements and methods for solving the tasks. They can be fixed by using better and shorter methods.

Appendixes
Class A

Denis Trenchev

```
=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[1].to_s.length == 5
                        arr1[i] = []
                        arr[i][0] = name[0]
                        arr[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[1].to_s.length == 5
                        arr1[i] = []
                        arr[i][0] = name_1[0]
                        arr[i][1] = name_1[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("students.csv", "w") do |csv|
  sort.each do |element|
    csv << element
  end
end
```

- The main problem is that the name of the CSV file should be “result.csv” instead of “students.csv”. In the Dir.glob-s is missing “.each” after the folder destination. The name of the array isn’t right when we’re searching for students it’s “arr” instead of the defined “arr1”. The program isn’t checking after the second underline is it a digit, or not.
- The quickest solution is by renaming the output CSV file and fixing the array names in the iteration loop, adding “.each” after the Dir.glob-s and adding checker for the digit in the name, to see if it’s the file name in the right form.
- I’m giving rank 1 for the program. The code can be easily read, but there are few mistakes. First of all the program won’t compile, because of errors made by inattention.

Marian Belchev

=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("_", 2).last.split("_").first

                number1 = file1.split("_").last.split(".").first


                firstName2      = file2.split("/").last.split("_").first

                lastName2       = file2.split("/").last.split("_", 2).last.split("_").first

                number2 = file2.split("_").last.split(".").first


                hash1[firstName1] = lastName1 + "." + number1

                hash2[firstName2] = lastName2 + "." + number2

        end

end


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

        hash2.sort.each do |key, value|

                if !hash1.has_key?(key) && !hash1.has_value?(value.split(".").first) &&
!hash1.has_value?(value.split(".").last.to_i)
```

```
                    csv << [key,value.gsub('.',"")]
            end
            if hash1.has_key?(key) && !hash1.has_value?(value.split(".").first) &&
!hash1.has_value?(value.split(".").last.to_i)
                csv << [key,value.gsub('.',"")]
            end
        end
end
```

- The main problem is in the name of the output CSV file, it should be “result.csv” instead of “results.csv”. An other problem is not recommended way to add the last names of the files to the hash with the first name, because the output in the csv won’t right.
- The quickest solution for solving the task is renaming the output CSV file and changing a little bit the adding way the last names to the hash.
- I’m giving rank 4 for the program. The program should work properly in the most cases after few changes.

Ivelin Slavchev

```
=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 != digit) 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
```

- The problem in the program is that in the iteration of the second folder, in the format check "if (ext2 != "rb") or (digit2.to_i.to_s != digit) or (short2.scan("_").count != 2)" should be "!= digit2". The other problem is in the sorting, because he didn't sorted by name when the length is equal.
- The solution is very easy, only fixing the variable name and adding double sort and I think it'll work for the most cases.
- I'll give rank 5, because there are 2 very little mistakes made by inattention.

Stanislav Valkanov

```
#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
```

- The problem in the program is only in the calculating the length of the second name. The other thing that can be an issue is that in the program doesn’t have check for the digit after the second underline.
- The solution is very quick, just change “last.to_s” with “last = last.to_s” and to add checking statement for the digit after the second underline and the program should work with the most cases.
- I’m giving rank 2, because the style of the code isn’t good, because the code can’t be read easily. I think that is very important the style of the code, because it should be written easy to read.

Petko Bozhinov

```
# 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
```

- The only problem is that in the checking statement for the digit should be “if file[2].to_s.numeric?” instead of “if file[2].numeric?”.
- The quickest solution for the task is adding “.to_s” in the checking statement for digit after the second underline.
- I’ll give rank 5, because the error is insignificant. The style of code is very good, easy readable and the program should work for the most of the cases.

Nikola Marinov

```
=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


requre '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
```

- The program is very different from the task requirement.
- To fix the program won’t be quick, because the mistake is huge. The smallest problem is the output CSV file name, it should be “result.csv” instead of “task.csv”. To fix the program we have to make big changes to the code. First of all we don’t need format check statement, we need to checking statement for digits in the name. Then we have

to change the checking length method, because we have to put the length divided by 2 and rounded to the smallest number.
- I'll give rank 1 to the code, because the code isn't easy readable, isn't right, isn't reliable and it definitely won't work.

Radoslav Kostadinov

=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|

                    s = my_text_file.split(/\//).last.capitalize

                    first_name = my_text_file.split("/").last.split("_").first

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




                    if s.count('_') == 2 and !((first_name == "" || first_name == " ") || (last_name ==
"" || last_name == " "))

                                        file1[first_name] = last_name

                              end

end


Dir.glob("#{path2}*.rb") do |my_text_file|

                    s = my_text_file.split(/\//).last.capitalize

                    first_name = my_text_file.split("/").last.split("_").first

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


                    if s.count('_') == 2 and !((first_name == "" || first_name == " ") || (last_name ==
"" || last_name == " "))

                                        file2[first_name] = last_name
                    end
end


CSV.open("result.csv", "w") do |csv|
        file1.sort.each do |first_name, last_name|
                file2.sort.each do |first_name1, last_name1|
                 if first_name1 == first_name and last_name1 == last_name
                    begin
                    end
                    else
                        csv << [last_name1, first_name1]
                    end
                end
        end
        end
```

- The first problem in the program is that the program don't have to capitalize the names of the files, it should be key sensitive. The second problem are the loops, which are checking for matches in the bot folders, totally wrong.
- The problem with the capitalize will be easily removed. The problem with the matching system will be solved by replacing the loops for the matching check in the end of the code with subtracting the hashes.
- I'll give rank 2, because it's readable. The mistake with the capitalize isn't big, but the mistake with the loops is kind of a big mistake.

Simeon Shopkin

```
=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
```

- There are mistakes when adding the files to array, in this way we can’t sort by length. The program should be key sensitive, so the capitalizes are useless and they’re changing the program work. The program is iterating second folder, but isn’t writing the files to an array.
- The solution isn’t quick. We have to change big part of the program. We have to add checking statement for the second folder files, changing the method of adding the file names to the array, removing the capitalizes and adding the sorting statement.
- I’ll give rank 1, because the program is totally wrong and won’t work with anything.

Kristina Pironkova

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

        if first_name.length == 10
```

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

end



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

    csv << [last,first]

    end
end
```

- The first problem is the name of the output CSV file. The program doesn't check for the form of the files, only check the length of the string before underline. The split "last_name" isn't right and the program should be key sensitive, so these capitalizes are useless and they're pushing the program not to work properly and the sort isn't sorting properly.
- The solution is to add checking statement for the format of the file, change the splits and remove the capitalizes. The name of the output CSV file should be "result.csv" instead of "results.csv". The sorting statement should be different, have to sort by last name DESC.
- I'll give rank 3, after few corrections the program should work properly in the most of the cases and the code is easy readable.

Ivo Valchev

```
=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("_")
                    isNum = Integer(name[2]) rescue nil
                    if name[0] and name[1] and name[0].length == 5 and !isNum!=nil
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("_")
                    isNum = Integer(name[2]) rescue nil
                    if name[0] and name[1] and name[0].length == 5 and !isNum!=nil
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
```

- The problem is in the checking statement in the iteration loops for the both folders.
- The solution for the problem is very quick, just changing “if name[0] and name[1] and name[0].length == 5 and !isNum!=nil hash_fold1.include?(name[0])” to “if name[0] and name[1] and name[0].length == 5 and !isNum!=nil and !hash_fold1.include?(name[0])” and changing “if name[0] and name[1] and name[0].length == 5 and !isNum!=nil and!hash_fold2.include?(name[0])” to “if name[0] and name[1] and name[0].length == 5 and !isNum!=nil and !hash_fold2.include?(name[0])”.
- I’ll give rank 3. The program will work with the most of the cases, it’s reliable, but it isn’t easy readable.

Dimitar Nestorov

```
=begin
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
=end

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| -el[1]}
CSV.open("result.csv", "w") do |csv|

	array.uniq.each do |e|

		csv << e

	end
end
```

- There is an error during sorting the array. Also sorting by last name isn`t reversed. There is a blank space after the comma between first name and last name.
- The first error is the presence of "-" before "el[1]". And also should be added ".reverese" after "sort_by {…}" and to be removed the empty space before last name.
- I'm giving rank 4 for the program. The program should work properly after 3 little changes. The program is easy to understand.

Lubomir Yankov

```
=begin
Develop a program named FirstName_LastName_ClassNumber_650c0b.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
=end

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("results.csv", "w") do |csv|

        array.each do |element|

                csv << element

        end

end
```

-The main problem is that the name of the CSV file should be “result.csv” instead of “results.csv”. There aren`t two arguments for the folders with files (there is only 1 agrument). Cheks whether the number of digits is 9, not 7. It doesn`t check if there is such a file in the array.
- The name of CSV file must be changed to “result.csv” and should be made a loop which counts from 0 to 1 (for the two folders) . The program must look for file names with 7 digits. Must be checked if already exists the same file into da array.
- I'm giving rank 3 for the program. The program should work properly after 4 changes.

Tihomir Lidanski

```
=begin
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

=end

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
```

- This code is not complete. Nothing puts in csv file and the program is not even close to the condition! Used the mod instead div.
- There is no quickest solution, you just have to write the program.
- I'm giving rank 1 for the program, because it`s not complete.

Kalin Marinov

```
=begin
Develop a program named FirstName_LastName_ClassNumber_bce70c.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'

hash = Hash.new

Dir.glob("#{ ARGV[0] }/*") do |name|
	name = name.split("/").last
	short_name = name.split('_')[1]
	if short_name.length == 5
		hash[name] = short_name
	end
end

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

- The program`s name isn`t correct! The name of the program should be "Kalin_Marinov_14_12_bce70c.rb" instead of "Kalin_Marinov_12_bce70c.rb".
Like a value in hash is put only last name, not the first name and last name. In CSV file is put the

key. The key is the name of the files, but the requested key is first name and last name, not full name of the files.
- The quickest solution for solving the task is renaming the name of the program and changing a little bit the adding way the first names to the hash too. And must be put value from hash, not the key.
- I'm giving rank 3 for the program. The program should work properly after 4-5 changes. The program is easy to understand.

Stanislav Gospodinov

```
=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("results.csv", "w") do |csv|
        hash.each do |key, value|
                csv << [key, value].flatten
        end
end
```

- Main problem is the name of the output CSV file, it should be "result.csv" instead of "results.csv". Used from key in hash is first name. This is bad, because if we have 2 or more people with same names, only one will be recorded in the hash.
- The quickest solution for solving the task is renaming the output CSV file. To fix the problem it can be used only the value from hash in the CSV file and then the key can take the program`s full name.
- I'm giving rank 3 for the program. The program should work properly after few changes. The program is easy to understand.

Borislav Rusinov

```
=begin
Develop a program named FirstName_LastName_ClassNumber_6fb3ad.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

a=ARGV[0]
require 'csv'
array=[]
Dir.glob("#{a}*.*")  do |my_text_file|
        name = my_text_file.split("/").last.split(".").first.split("_")
        if name[1]!=nil && name[0].length==10
                array << name[0] + "," + name[1]
        end
end
array.sort!
array.reverse!
```

```
File.open("results.csv", "w") do |csv|
	array.each do |arg|
	csv.puts(arg)
	end
end
```

- To be changed the name of CSV file to “result.csv” instead of “results.csv”. The array is sorted by first name instead of last name.
- These errors could be eliminated with the changing of CSV file`s name and in the array first be imported last name , then first name (with comma between them) and after that the elements of the array must be splitted by comma and be put in CVS file – first name, then last name.

```
first = arg.split(",")[1]
second = arg.split(",")[0]
csv.puts("#{first},#{second}")
```

- I'm giving rank 3 for the program. The program should work properly after few changes. The program is easy to understand.

Momchil Angelov

```
=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= short.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
```

- In the conditio of two "if" is applied "=". It doesn`t check if between the digit and the extension there is something else. The sorting isn`t correct.
- The quickest solution for solving the task is first the conditions of the two if-s must be "==", instead of "=". The sorting must be "arr3 = arr3.sort_by{|el| [el[1], el[0]]}"
- I'm giving rank 3 for the program. The program should work properly after 3 changes.
The program is not easy to understand.

Veselin Dechev

```
=begin
Develop a program named FirstName_LastName_ClassNumber_5f1c22.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'
result = Hash.new
Dir.glob(ARGV[0] + "*.rb").each do |first|
	name1 = first.split("/").last.capitalize
	first_name = name1.split("_").first.capitalize
	last_name = name1.split("_",2).last.split('_').first.capitalize
	Dir.glob(ARGV[1]+"*.rb").each do |second|
		name2 = second.split("/").last.capitalize
		if (name1 == name2)
			result.compare_by_identity
			result[first_name] = last_name
		end
	end
end
CSV.open("result.csv", "w") do |csv|
	result.sort_by{|k, v| k}.each do |element|
		csv << element
	end
end
```

- The name of the program is incorrect and must be changed to “Veselin_Dechev_11_5f1c22.rb”, instead of “Veselin_Dechev_11A2_5f1c22.rb”. In the condition is searched sameness in files by first name and last name, not by full program`s name. The key from “result (hash)” is the first name and this may lead to lack of student`s name in the CSV file. The condition requires the output to be sorted by last name, not by first name.
- Must be changed the name of the program in the correct format. Must be created first name and last name for both folders and after their comparison if they are different to be put in the first name and the last name of the student from the first folder and accordingly “result (hash)” to be sorted by value with the beginning of value – last name The key of “result (hash)” must be the full name of the program, not only the first name.
- I'm giving rank 2 for the program. The program should work properly after many changes.

Dimitar Terziev

```
=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(' ')
	}
}
```

- It isn`t necessary: file_str=~/\A[a-zA-Z]+_[a-zA-Z]+_\d+\.rb\z/
- Nothing should be corrected.
- I'm giving rank 5 for the program. Good job!

Class B

Borislav Stratev

```
#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
```

- The created file is named “results.csv”. Also, the program does not check for correct formatting of the file names. Incorrect comparing.

- The quickest solution is to change the output file name to “result.csv”. Other mistakes can be fixed by adding a check for file names.

The comparing statement should use the “subtract” method.

- I rank 3 out of 5, because the code is readable, but does not work.

David Georgiev

```
#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(/_/)

          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
```

- The code does not compile, because of an undefined method “length”. Also it does not have a checking statement for correct file name format.

- To fix this the variable in the “if” statement must be converted to string and a file name format check must be added.

- I rank 4 out of 5. The code can be shortened and quickened, but is clean and readable.

Iliyan Germanov

```
=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
```

- The code is working properly in most cases. Problems arise with funky file names.

- Can be improved by adding more checking statements.

- I rank 5 out of 5. The code is readable and correct.

Lili Kokalova

```
=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
```

- Main error is the check for difference, which does not work.

- Can be fixed by using the “subtract” method.

- I rank 4 out 5. Easy to read code, but with small incorrections.

Nikolay Mihailov

```
#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
```

- The program experiences an error when compiling, because of an undefined “each” method.

- A solution is to add a “split” method for the string.

- I rank 4 out of 5, because the code is readable, but does not compile.

Stanislav Iliev

```
#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
```

- The code compiles, but the check statement is incorrect. Also the program lacks format checking and does not sort the hash.

- Can be fixed by using the “substract” method on the arrays, adding format checking and a sort method.

- I rank 3 out of 5. The code is hard to understand.

Stefan Iliev

```
#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
```

- The program creates a file named “results.csv”.

- The .csv file must be named “result.csv”.

- I rank 4 out of 5. Working, but complicated code.

Valentin Varbanov

```
=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
```

- Error when compiling the code, because of a local variable. Missing “require 'csv' ” and a sort method.

- Add missing libraries and sort the hash.
- I rank 2 out of 5, because of the errors in the program.

Veselina Kolova

```
=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) 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("_")[1]] = text_file.split("_")[0]

                    end

                end

        end

end


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


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

  people.each do |element|

  csv << element

  end

end
```

- The code does not compile because of syntax errors – unexpected identifier and a missing keyword “end”.

- The solution is to fix the syntax errors.
- I rank 3 out of 5. The code is understandable, but has many errors.

Vladimir Yordanov

```
#Develop a program named FirstName_LastName_ClassNumber_4bbed0.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




names = Hash.new

Dir.glob (ARGV[0] + "*.rb") do |file|
        if (ARGV[1] == true)
                ARGV[1] == false
        end

        slice = file.split("/").last
        first_name = slice.split('_')[0]
        second_name = slice.split('_')[1]
        if (second_name.length == 5)
                #print first_name
                #puts second_name
                names[first_name] = second_name
        end

end

names = names.sort
puts names

require 'csv'
CSV.open("results.csv", "w") do |csv|
        names.to_a.each do |element|
                csv << element
        end
end
```

- The program does not compile correctly, because of an undefined method “length” in the “if” statement. The sort method used is incorrect.

- Can be fixed by converting the variable to string before comparing and by fixing the sort method.

- I rank 3 out of 5. The code is easy to understand, but has many errors.